

**IMPROVED COMPUTING ARCHITECTURE AND RELATED SYSTEM AND METHOD**

Publication number: JP2006518495 (T)

Publication date: 2006-06-10

Inventor(s):

Applicant(s):

Classification:





- international: G06F15/80; G06F9/30; G06F9/38; G06F9/445; G06F9/46; G06F15/78; G06F15/76; G06F9/30; G06F9/38; G06F9/445; G06F9/46

- European: G06F9/3884

Application number: JP20050502226T 20031031

Priority number(s): US20030422503P 20021031; US20030684102 20031009; US20030683929 20031009; US20030683932 20031009; US20030684053 20031009; US20030684057 20031009; WO20030634559 20031031

Also published as:

 WO2004042560 (A2) WO2004042560 (A3) WO2004042574 (A2) WO2004042574 (A3) WO2004042569 (A2)

more &gt;&gt;

Abstract not available for JP 2006518495 (T)

Abstract of corresponding document: **WO 2004042560 (A2)**

A peer-vector machine includes a host processor and a hardwired pipeline accelerator. The host processor executes a program, and, in response to the program, generates host data, and the pipeline accelerator generates pipeline data from the host data. Alternatively, the pipeline accelerator generates the pipeline data, and the host processor generates the host data from the pipeline data. Because the peer-vector machine includes both a processor and a pipeline accelerator, it can often process data more efficiently than a machine that includes only processors or only accelerators. For example, one can design the peer-vector machine so that the host processor performs decision-making and non-mathematically intensive operations and the accelerator performs non-decision-making and mathematically intensive operations. By shifting the mathematically intensive operations to the accelerator, the peer-vector machine often can, for a given clock frequency, process data at a speed that surpasses the speed at which a processor-only machine can process the data.

Data supplied from the **esp@cenet** database — Worldwide

(19) 日本国特許庁(JP)

(12) 公表特許公報(A)

(11) 特許出願公表番号

特表2006-518495

(P2006-518495A)

(43) 公表日 平成18年8月10日(2006.8.10)

(51) Int. Cl.

G06F 15/00 (2006.01)

F1

G06F 15/00

テーマコード(参考)

審査請求 未請求 予備審査請求 未請求 (全 33 頁)

(21) 出願番号	特願2005-502226 (P2006-502226)	(71) 出願人	504242618
(86) (22) 出願日	平成15年10月31日(2003.10.31)		ロッキード マーティン コーポレーション
(85) 翻訳文提出日	平成17年6月16日(2005.6.16)		アメリカ合衆国 メリーランド州 208
(86) 国際出願番号	PCT/US2003/034559		17-1803 ベセスダ ロックレッズ
(87) 国際公開番号	W02004/042574		ドライブ 6801
(87) 国際公開日	平成16年5月21日(2004.5.21)	(74) 代理人	100083932
(31) 優先権主張番号	60/422,503		弁理士 廣江 武典
(32) 優先日	平成14年10月31日(2002.10.31)	(74) 代理人	100129698
(33) 優先権主張国	米国(US)		弁理士 武川 隆寛
(31) 優先権主張番号	10/684,102	(74) 代理人	100129676
(32) 優先日	平成15年10月9日(2003.10.9)		弁理士 ▲高▼荒 新一
(33) 優先権主張国	米国(US)	(74) 代理人	100130074
(31) 優先権主張番号	10/683,929		弁理士 中村 繁元
(32) 優先日	平成15年10月9日(2003.10.9)		
(33) 優先権主張国	米国(US)		

最終頁に続く

(54) 【発明の名称】 改善された計算アーキテクチャを有する計算マシン、関連システム、並びに、方法

## (57) 【要約】

計算マシンは第1バッファと該バッファと結合されたプロセッサとを含む。プロセッサは、アプリケーション、第1データ転送オブジェクト、並びに、第2データ転送オブジェクトを実行し、アプリケーションの制御下でデータを発行し、第1データ転送オブジェクトの制御下でその発行されたデータをバッファにロードし、そして、第2データ転送オブジェクトの制御下でバッファからその発行されたデータを検索する。代替的には、プロセッサは、データを検索して、第1データ転送オブジェクトの制御下でその検索されたデータをバッファにロードし、第2データ転送オブジェクトの制御下でバッファからそのデータをアンロードし、そして、アプリケーションの制御下でそのアンロードされたデータを処理する。計算マシンがプロセッサと結合されたハードウェアに組み込まれたパイプライン加速器を含むビークトル・マシンである場合、バッファ及びデータ転送オブジェクトはアプリケーション及び加速器の間でのデータの転送を補助する。

10

## 【特許請求の範囲】

## 【請求項1】

計算マシンであって、

第1バッファと、

前記バッファと結合されたプロセッサと、を含み、

前記プロセッサが、

アプリケーション、第1データ転送オブジェクト、並びに、第2データ転送オブジェクトを実行し、

前記アプリケーションの制御下でデータを発行し、

前記第1データ転送オブジェクトの制御下で前記発行されたデータを前記バッファにロードし、

前記第2データ転送オブジェクトの制御下で前記バッファから前記発行されたデータを検索するように動作できる、計算マシン。

## 【請求項2】

前記第1及び第2のデータ転送オブジェクトが、同一オブジェクト・コードの第1及び第2の例証をそれぞれ含む、請求項1に記載の計算マシン。

## 【請求項3】

前記プロセッサが、

前記アプリケーションを実行して、前記アプリケーションの制御下で前記データを発行するように動作できる処理ユニットと、

前記第1及び第2のデータ転送オブジェクトを実行して、前記第1データ転送オブジェクトの制御下で前記バッファに前記発行されたデータをロードし、前記第2データ転送オブジェクトの制御下で前記発行されたデータを検索するように動作できるデータ転送ハンドラーを含む、請求項1に記載の計算マシン。

## 【請求項4】

前記プロセッサが、前記アプリケーションのスレッドを実行して、前記スレッドの制御下で前記データを発行するように更に動作できる、請求項1に記載の計算マシン。

## 【請求項5】

前記プロセッサが、

キュー・オブジェクト及びリーダー・オブジェクトを実行し、

前記キュー・オブジェクトの制御下で前記発行されたデータの前記バッファへのローディングを反映しているキュー値を記憶し、

前記リーダー・オブジェクトの制御下で前記キュー値を読み取り、

前記リーダー・オブジェクトの制御下で且つ前記キュー値に応じて、前記発行されたデータが前記バッファを専有していることを前記第2ソフトウェア・オブジェクトに通知し、

前記第2データ転送オブジェクトの制御下で且つ前記通知に応じて、前記記憶箇所から前記発行されたデータを検索するように更に動作できる、請求項1に記載の計算マシン。

## 【請求項6】

バスを含み、

前記プロセッサが通信オブジェクトを実行し、前記通信オブジェクトの制御下で前記検索されたデータを前記バスに駆動するように動作できる、請求項1に記載の計算マシン。

## 【請求項7】

第2バッファを含み、

前記プロセッサが、前記第2データ転送オブジェクトの制御下で、前記検索されたデータを前記第2バッファに提供するように動作できる、請求項1に記載の計算マシン。

## 【請求項8】

前記プロセッサが、前記第2データ転送オブジェクトの制御下で、ヘッダーと前記検索されたデータとを含むメッセージを生成するように更に動作できる、請求項1に記載の計算マシン。

## 【請求項9】

前記第1及び第2のデータ転送オブジェクトが同一オブジェクト・コードの第1及び第2の例証をそれぞれ含み、

前記プロセッサがオブジェクト・ファクトリーを実行して、前記オブジェクト・ファクトリーの制御下で前記オブジェクト・コードを生成するように動作できる、請求項1に記載の計算マシン。

【請求項10】

計算マシンであって、

第1バッファと、

前記バッファと結合されたプロセッサと、を含み、

前記プロセッサが、

第1及び第2のデータ転送オブジェクトとアプリケーションとを実行し、

前記第1データ転送オブジェクトの制御下でデータを検索して前記検索されたデータを前記バッファにロードし、

前記第2データ転送オブジェクトの制御下で前記バッファから前記データをアンロードし、

前記アプリケーションの制御下で前記アンロードされたデータを処理するように動作できる、計算マシン。

【請求項11】

前記第1及び第2のデータ転送オブジェクトが同一オブジェクト・コードの第1及び第2の例証をそれぞれに含む、請求項10に記載の計算マシン。

【請求項12】

前記プロセッサが、

前記アプリケーションを実行し、前記アプリケーションの制御下で前記アンロードされたデータを処理するように動作できる処理ユニットと、

前記第1及び第2のデータ転送オブジェクトを実行し、前記第1データ転送オブジェクトの制御下で前記バスから前記データを検索して該データを前記バッファにロードし、前記第2データ転送オブジェクトの制御下で前記バッファから前記データをアンロードするように動作できるデータ転送ハンドラーと、を含む、請求項10に記載の計算マシン。

【請求項13】

前記プロセッサが前記アプリケーションのスレッドを実行して、前記スレッドの制御下で前記アンロードされたデータを処理するように更に動作できる、請求項10に記載の計算マシン。

【請求項14】

前記プロセッサが、

キュー・オブジェクト及びリーダー・オブジェクトを実行し、

前記キュー・オブジェクトの制御下で前記発行されたデータの前記第1バッファへのローディングを反映しているキュー値を記憶し、

前記リーダー・オブジェクトの制御下で前記キュー値を読み取り、

前記リーダー・オブジェクトの制御下で且つ前記キュー値に応じて、前記発行されたデータが前記バッファを占有していることを前記第2データ転送オブジェクトに通知し、

前記第2データ転送オブジェクトの制御下で且つ前記通知に応じて前記バッファから前記発行されたデータをアンロードするように更に動作できる、請求項10に記載の計算マシン。

【請求項15】

第2バッファを更に含み、

前記プロセッサが前記第1データ転送オブジェクトの制御下で前記第2バッファから前記データを検索するように動作できる、請求項10に記載の計算マシン。

【請求項16】

バスを更に含み、

前記プロセッサが、通信オブジェクトを実行し、前記通信オブジェクトの制御下で前記

バスから前記データを検索して、前記第1データ転送オブジェクトの制御下で前記通信オブジェクトから前記データを検索するように動作できる、請求項10に記載の計算マシン。

【請求項17】

前記第1及び第2のデータ転送オブジェクトが同一のオブジェクト・コードの第1及び第2の例証をそれぞれ含み、

前記プロセッサが、オブジェクト・ファクトリーを実行して、前記オブジェクト・ファクトリーの制御下で前記オブジェクト・コードを生成するように動作できる、請求項10に記載の計算マシン。

【請求項18】

前記プロセッサが、前記第1データ転送オブジェクトの制御下でヘッダー及び前記データを含むメッセージから前記データを回復するように動作できる、請求項10に記載の計算マシン。

【請求項19】

ビアーベクトル・マシンであって  
バッファと、  
バスと、

前記バッファ及び前記バスの双方と結合されたプロセッサであり、

アプリケーション、第1及び第2のデータ転送オブジェクト、並びに、通信オブジェクトを実行し、

前記アプリケーションの制御下でデータを発行し、

前記第1データ転送オブジェクトの制御下で前記バッファに前記発行されたデータをロードし、

前記第2データ転送オブジェクトの制御下で前記バッファから前記発行されたデータを検索し、

前記通信オブジェクトの制御下で前記発行されたデータを前記バスに駆動するように動作できるプロセッサと、

前記バスと結合されて、前記バスから前記発行されたデータを受信して前記受信された発行データを処理するように動作できるパイプライン加速器と、を含むビアーベクトル・マシン。

【請求項20】

前記プロセッサが、前記第2データ転送オブジェクトの制御下で前記発行されたデータを含むメッセージを構築して、前記通信オブジェクトの制御下で前記バスに前記メッセージを駆動するように更に動作でき、

前記パイプライン加速器が、前記バスから前記メッセージを受信して、前記メッセージから前記発行されたデータを回復するように動作できる、請求項19に記載のビアーベクトル・マシン。

【請求項21】

前記ホストプロセッサと結合されると共にオブジェクト・データを記憶するように動作できるレジストリを更に含み、

前記プロセッサが、

オブジェクト・ファクトリーを実行し、

前記オブジェクト・ファクトリーの制御下で、前記オブジェクト・データから、前記第1及び第2のデータ転送オブジェクトと前記通信オブジェクトとを生成するように動作できる、請求項19に記載のビアーベクトル・マシン。

【請求項22】

ビアーベクトル・マシンであって、  
バッファと、  
バスと、

前記バスと結合されると共にデータを生成してそのデータを前記バスに駆動するように

動作できるパイプライン加速器と、

前記バッファ及び前記バスの双方と結合されたプロセッサと、を含み、

前記プロセッサが、

アプリケーション、第1及び第2のデータ転送オブジェクト、並びに、通信オブジェクトを実行し、

前記通信オブジェクトの制御下で前記バスから前記データを受信し、

前記第1データ転送オブジェクトの制御下で前記受信したデータを前記バッファにロードし、

前記第2データ転送オブジェクトの制御下で前記バッファから前記データをアンロードし、

前記アプリケーションの制御下で前記アンロードされたデータを処理するように動作できる、ビアーベクトル・マシン。

【請求項23】

前記パイプライン加速器が前記データを含むメッセージを構築してそのメッセージを前記バスに駆動するように更に動作でき、

前記プロセッサが、

前記通信オブジェクトの制御下で前記バスから前記メッセージを受信し、

前記第1データ転送オブジェクトの制御下で前記メッセージから前記データを回復するように動作できる、請求項22に記載のビアーベクトル・マシン。

【請求項24】

前記ホストプロセッサと結合されると共にオブジェクト・データを記憶するように動作できるレジストリを更に含み、

前記プロセッサが、

オブジェクト・ファクトリーを実行し、

前記オブジェクト・ファクトリーの制御下で、前記オブジェクト・データから前記第1及び第2のデータ転送オブジェクトと前記通信オブジェクトを生成するように動作できる、請求項22に記載のビアーベクトル・マシン。

【請求項25】

ビアーベクトル・マシンであって、

第1バッファと、

バスと、

前記バッファ及び前記バスの双方と結合されたプロセッサであり、

コンフィギュレーション・マネージャ、第1及び第2のデータ転送オブジェクト、並びに、通信オブジェクトを実行し、

前記コンフィギュレーション・マネージャ及び前記第1データ転送オブジェクトの制御下でコンフィギュレーション・ファームウェアを前記バッファにロードし、

前記第2データ転送オブジェクトの制御下で前記バッファから前記コンフィギュレーション・ファームウェアを検索し、

前記通信オブジェクトの制御下で前記コンフィギュレーション・ファームウェアを前記バスに駆動するように動作できるプロセッサと、

前記バスと結合されると共に、前記コンフィギュレーション・ファームウェアを受信して、それ自体をそのコンフィギュレーション・ファームウェアで構成するように動作できるパイプライン加速器と、

を含むビアーベクトル・マシン。

【請求項26】

前記プロセッサが、前記第2データ転送オブジェクトの制御下で前記コンフィギュレーション・ファームウェアを含むメッセージを構築して、前記通信オブジェクトの制御下でそのメッセージを前記バスに駆動するように更に動作でき、

前記パイプライン加速器が前記バスから前記メッセージを受信して、そのメッセージから前記コンフィギュレーション・ファームウェアを回復するように動作できる。請求項2

10

20

30

40

50

5 に記載のビアーベクトル・マシン。

【請求項 27】

前記プロセッサと結合されると共にコンフィギュレーション・データを記憶するように動作できるレジストリーを更に含み、

前記プロセッサが前記コンフィギュレーション・マネージャの制御下で前記コンフィギュレーション・データから前記コンフィギュレーション・ファームウェアを位置決めするように動作できる、請求項 25 に記載のビアーベクトル・マシン。

【請求項 28】

第 2 バッファを更に含み、

前記プロセッサが、

アプリケーションと第 3 及び第 4 のデータ転送オブジェクトとを実行し、

前記コンフィギュレーション・マネージャの制御下でコンフィギュレーション命令を生成し、

前記第 3 データ転送オブジェクトの制御下で前記コンフィギュレーション命令を前記第 2 バッファにロードし、

前記第 4 データ転送オブジェクトの制御下で前記第 2 バッファから前記コンフィギュレーション命令を検索し、

前記アプリケーションの制御下で、前記コンフィギュレーション命令と対応している動作を実行すべく前記アプリケーションを構成するように動作できる、請求項 25 に記載のビアーベクトル・マシン。

【請求項 29】

前記プロセッサが、

前記コンフィギュレーション・マネージャの制御下でコンフィギュレーション命令を生成し、

前記アプリケーションの制御下で、前記コンフィギュレーション命令と対応している動作を実行すべく前記アプリケーションを構成するように動作できる、請求項 25 に記載のビアーベクトル・マシン。

【請求項 30】

前記コンフィギュレーション・マネージャが、前記ファームウェアをロードする前に、前記パイプライン加速器が前記コンフィギュレーション・データによって規定されるコンフィギュレーションを支援することを確認するように動作できる、請求項 25 に記載のビアーベクトル・マシン。

【請求項 31】

ビアーベクトル・マシンであって、

第 1 バッファと、

バスと、

前記バスと結合されると共に、例外データを生成して、その例外データを前記バスに駆動するように動作できるパイプライン加速器と、

前記バッファ及び前記バスの双方と結合されたプロセッサと、を含み、

前記プロセッサが、

例外マネージャ、第 1 及び第 2 のデータ転送オブジェクト、並びに、通信オブジェクトを実行し、

前記通信オブジェクトの制御下で前記バスから前記例外データを受信し、

前記第 1 データ転送オブジェクトの制御下で前記受信された例外データを前記バッファにロードし、

前記第 2 データ転送オブジェクトの制御下で前記バッファから前記例外データをアンロードし、

前記例外マネージャの制御下で前記アンロードされた例外データを処理するように動作できる、ビアーベクトル・マシン。

【請求項 32】

前記パイプラインが、前記例外データを含むメッセージを構築して、そのメッセージを前記バスに駆動するように更に動作でき、

前記プロセッサが、前記通信オブジェクトの制御下で前記バスから前記メッセージを受信して、前記第1データ転送オブジェクトの制御下で前記メッセージから前記例外データを回復するように動作できる、請求項31に記載のビアーベクトル・マシン。

【請求項33】

第2バッファを更に含み、

前記プロセッサが、

コンフィギュレーション・マネージャ及び、第3及び第4データ転送オブジェクトを実行し、

前記例外データに応じて、前記コンフィギュレーション・マネージャの制御下でコンフィギュレーション・ファームウェアを生成し、

前記第3データ転送オブジェクトの制御下で前記コンフィギュレーション・ファームウェアを前記第2バッファにロードし、

前記第4データ転送オブジェクトの制御下で前記第2バッファから前記コンフィギュレーション命令をアンロードし、

前記通信オブジェクトの制御下で前記コンフィギュレーション・ファームウェアを前記バスに駆動するように更に動作でき、

前記パイプライン加速器が、前記バスから前記コンフィギュレーション・ファームウェアを受信して、それ自体を該ファームウェアで再構成するように動作できる、請求項31に記載のビアーベクトル・マシン。

【請求項34】

前記プロセッサが、

アプリケーション及びコンフィギュレーション・マネージャを実行し、

前記例外データに応じて、前記コンフィギュレーション・マネージャの制御下でコンフィギュレーション命令を生成し、

前記コンフィギュレーション命令に応じて、前記アプリケーションの制御下で前記アプリケーションを再構成するように更に動作できる、請求項31に記載のビアーベクトル・マシン。

【請求項35】

ビアーベクトル・マシンであって、

コンフィギュレーション・データを記憶するように動作できるコンフィギュレーション・レジストリと、

前記コンフィギュレーション・レジストリと結合されると共に、前記コンフィギュレーション・データからコンフィギュレーション・ファームウェアを位置決めするように動作できるプロセッサと、

前記プロセッサと結合されると共に、前記コンフィギュレーション・ファームウェアでそれ自体を構成するように動作できるパイプライン加速器と、  
を含むビアーベクトル・マシン。

【請求項36】

ビアーベクトル・マシンであって、

コンフィギュレーション・データを記憶するように動作できるコンフィギュレーション・レジストリと、

パイプライン加速器と、

前記コンフィギュレーション・レジストリ及び前記パイプライン加速器の双方と結合されると共に、前記コンフィギュレーション・データに応じてコンフィギュレーション・ファームウェアを検索して、そのコンフィギュレーション・ファームウェアで前記パイプライン加速器を構成するように動作できるプロセッサと、  
を含むビアーベクトル・マシン。

【請求項37】



方法であって、  
 アプリケーションでデータを発行し、  
 第1データ転送オブジェクトで前記発行されたデータを第1バッファにロードし、  
 第2データ転送オブジェクトで前記バッファから前記発行されたデータを検索すること  
 を含む方法。

【請求項38】

前記データを発行することが、前記アプリケーションのスレッドで前記データを発行することを含む、請求項37に記載の方法。

【請求項39】

前記バッファにおける前記発行されたデータの存在と対応するキュー値を生成し、  
 前記キュー値に応じて、前記発行されたデータが前記バッファを専有していることを前  
 記第2データ転送オブジェクトに通知することを更に含む、  
 前記発行されたデータを検索することが、前記通知に応じて、前記第2データ転送オブ  
 ジェクトで前記記憶箇所から前記発行されたデータを検索することを含む、請求項37に  
 記載の方法。

【請求項40】

通信オブジェクトで前記検索されたデータをバスに駆動することを更に含む、請求項37に記載の方法。

【請求項41】

前記第2データ転送オブジェクトで前記検索されたデータを第2バスにロードすること  
 を更に含む、請求項37に記載の方法。

【請求項42】

前記第2データ転送オブジェクトで前記検索されたデータに対するヘッダーを生成し、  
 前記第2データ転送オブジェクトで前記ヘッダー及び前記検索されたデータを1つのメ  
 ッセージに組み合わせることを更に含む、請求項37に記載の方法。

【請求項43】

オブジェクト・ファクトリーでデータ転送オブジェクト・コードを生成し、  
 前記オブジェクト・コードの第1例証として前記第1データ転送オブジェクトを生成し、  
 前記オブジェクト・コードの第2例証として前記第2データ転送オブジェクトを生成す  
 ることを更に含む、請求項37に記載の方法。

【請求項44】

パイプライン加速器で前記第2データ転送オブジェクトから前記データを受信して処理  
 することを更に含む、請求項37に記載の方法。

【請求項45】

方法であって、  
 第1データ転送オブジェクトで、データを検索して、その検索されたデータを第1バッ  
 ファにロードし、  
 第2データ転送オブジェクトで前記バッファから前記データをアンロードし、  
 アプリケーションで前記アンロードされたデータを処理することを含む方法。

【請求項46】

前記アンロードされたデータを処理することが、前記アプリケーションのスレッドで前  
 記アンロードされたデータを処理することを含む、請求項45に記載の方法。

【請求項47】

前記バッファにおける前記データの存在と対応するキュー値を生成し、  
 前記キュー値に応じて、前記データが前記バッファを専有していることを前記第2デー  
 タ転送オブジェクトに通知し、  
 前記データをアンロードすることが、前記通知に応じて、前記第1データ転送オブジェ  
 クトで前記バッファから前記データをアンロードすることを含む、請求項45に記載の方  
 法。

## 【請求項 48】

前記データを検索することが、前記第1データ転送オブジェクトで第2バッファから前記データを検索することを含む、請求項45に記載の方法。

## 【請求項 49】

通信オブジェクトでバスから前記データを受信することを更に含み、

前記データを検索することが、前記第1データ転送オブジェクトで前記通信オブジェクトから前記データを検索することを含む、請求項45に記載の方法。

## 【請求項 50】

パイプライン加速器で前記データを前記第1データ転送オブジェクトに提供することを更に含み、請求項45に記載の方法。

10

## 【請求項 51】

方法であって、

プロセッサ上で走っているアプリケーションでデータを発行し、

前記プロセッサ上で走っている第1データ転送オブジェクトで前記発行されたデータをバッファにロードし、

前記プロセッサ上で走っている第2データ転送オブジェクトで前記バッファから前記発行されたデータを検索し、

前記プロセッサ上で走っている通信オブジェクトで前記検索された発行データをバスに駆動し、

前記バスから前記発行されたデータを受信して、パイプライン加速器で前記発行されたデータを処理することを含む方法。

20

## 【請求項 52】

前記第2データ転送オブジェクトで、ヘッダーと前記発行されたデータとを含むメッセージを生成し、

前記データを前記バスに駆動することが、前記通信オブジェクトで前記メッセージを前記バスに駆動することを含み、

前記発行されたデータを受信して処理することが、前記パイプライン加速器で、前記メッセージを受信してそのメッセージから前記発行されたデータを回復することを含む、請求項51に記載の方法。

## 【請求項 53】

30

方法であって、

パイプライン加速器で、データを生成してそのデータをバスに駆動し、

通信オブジェクトで、前記バスから前記データを受信し、

第1データ転送オブジェクトで、前記受信されたデータをバッファにロードし、

第2データ転送オブジェクトで、前記バッファから前記データをアンロードし、

アプリケーションで、前記アンロードされたデータを処理することを含む方法。

## 【請求項 54】

前記データを生成することが、前記パイプライン加速器で、ヘッダー及び前記データを含むメッセージを構築することを含み、

前記データを駆動することが、前記パイプライン加速器で、前記メッセージを前記バスに駆動することを含み、

40

前記データを受信することが、前記通信オブジェクトで、前記バスから前記メッセージを受信することを含み、

前記第1データ転送オブジェクトで前記メッセージから前記データを回復することを含み、請求項53に記載の方法。

## 【請求項 55】

コンフィギュレーション・マネージャでコンフィギュレーション・ファームウェアを検索し、

第1通信オブジェクトで、前記コンフィギュレーション・ファームウェアを第1バッファにロードし、

50

第2通信オブジェクトで、前記バッファから前記コンフィギュレーション・ファームウェアを検索し、

通信オブジェクトで、前記コンフィギュレーション・ファームウェアをバスに駆動し、  
パイプライン加速器で、前記コンフィギュレーション・ファームウェアを受信し、  
前記コンフィギュレーション・ファームウェアで、前記パイプライン加速器を構成することを含む方法。

【請求項56】

前記コンフィギュレーション・マネージャでコンフィギュレーション命令を生成し、  
前記アプリケーションを構成して、前記コンフィギュレーション命令と対応している動作を実行することを更に含む、請求項55に記載の方法。

10

【請求項57】

前記コンフィギュレーション・マネージャでコンフィギュレーション命令を生成し、  
第3通信オブジェクトで、前記コンフィギュレーション命令を第2バッファにロードし、

第4通信オブジェクトで、前記第2バッファから前記コンフィギュレーション命令を検索し、

前記アプリケーションを構成して、前記コンフィギュレーション命令と対応している動作を実行することを更に含む、請求項55に記載の方法。

【請求項58】

方法であって、  
パイプライン加速器で、例外データを生成してその例外データをバスに駆動し、  
通信オブジェクトで、前記バスから前記例外データを受信し、  
第1データ転送オブジェクトで、前記受信された例外データをバッファにロードし、  
第2データ転送オブジェクトで、前記バッファから前記例外データをアンロードし、  
例外マネージャで、前記アンロードされた例外データを処理することを含む方法。

20

【請求項59】

前記例外データに応じて、コンフィギュレーション・マネージャでコンフィギュレーション・ファームウェアを検索し、

第3転送オブジェクトで、前記コンフィギュレーション・ファームウェアを第2バッファにロードし、

第4データ転送オブジェクトで、前記第2バッファから前記コンフィギュレーション命令をアンロードし、

前記通信オブジェクトで、前記コンフィギュレーション・ファームウェアを前記バスに駆動し、

前記コンフィギュレーション・ファームウェアで、前記パイプライン加速器を再構成することを更に含む、請求項58に記載の方法。

30

【請求項60】

エラー・データに応じて、コンフィギュレーション・マネージャでコンフィギュレーション命令を生成し、

前記コンフィギュレーション命令に応じて前記アプリケーションを再構成することを更に含む、請求項58に記載の方法。

40

【請求項61】

方法であって、  
計算マシンの初期化中、コンフィギュレーション・レジストリに記憶されたコンフィギュレーション・データによって指されたコンフィギュレーション・ファームウェアを検索し、

前記コンフィギュレーション・ファームウェアで、前記計算マシンのパイプライン加速器を構成することを含む方法。

【発明の詳細な説明】

【技術分野】

50

【0001】

## ＜優先権の請求＞

この出願は、下記の特許文献1に対する優先権を請求するものであり、引用することによってここに合体させる。

【特許文献1】米国仮出願第60/422,503号(2002年10月31日出願)

【0002】

## ＜関連出願の相互参照＞

この出願は、「改善された計算アーキテクチャ、関連システム、並びに、方法」と題された下記の特許文献2、「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された下記の特許文献3、「プログラマブル回路、関連計算マシン、並びに、方法」と題された下記の特許文献4、「多数パイプライン・ユニットを有するパイプライン加速器、関連計算マシン、並びに、方法」と題された下記の特許文献5、と関連し、これら特許文献は全て2003年10月9日出願され、共通の所有者を有し、引用することによってここに合体させる。

【特許文献2】米国出願第10/684,102号

【特許文献3】米国出願第10/683,929号

【特許文献4】米国出願第10/684,057号

【特許文献5】米国出願第10/683,932号

【背景技術】

【0003】

比較的大量のデータを比較的短い期間で処理する通常の計算アーキテクチャは、処理負担を分担する多数の相互接続プロセッサを含む。処理負担を分担することによって、これら多数のプロセッサは、しばしば、所与のクロック周波数で単一プロセッサができるものよりよりも迅速にデータを処理できる。例えば、これらプロセッサの各々はデータの各部分を処理できるか、或は、処理アルゴリズムの各部分を実行できる。

【0004】

図1は、多数プロセッサ・アーキテクチャを有する従来の計算マシン10の概略ブロック図である。この計算マシン10は、マスター・プロセッサ12と、相互に通信すると共に該マスター・プロセッサとバス16を介して通信する共同プロセッサ14、…14、と、遠隔装置(図1では不図示)から生データを受け取る入力ポート18と、該遠隔装置に処理データを提供する出力ポート20とを含む。また、計算マシン10はマスター・プロセッサ12に対するメモリ22と、共同プロセッサ14、…14に対する各メモリ24、…24、と、マスター・プロセッサ及び共同プロセッサがバス16を介して共有するメモリ26とを含む。メモリ22はマスター・プロセッサ12に対するプログラム及び作業メモリの双方の役割を果たし、各メモリ24、…24は各共同メモリ14、…14に対するプログラム及び作業メモリの双方の役割を果たす。共有されたメモリ26は、マスター・プロセッサ12及び共同プロセッサ14がそれらの間でデータを転送すること、ポート18を介して遠隔装置からデータを転送すること、ポート20を介して遠隔装置にデータを転送することを可能としている。またマスター・プロセッサ12及び共同プロセッサ14は、マシン10が生データを処理する速度を制御する共通クロック信号を受け取る。

【0005】

一般に、計算マシン10は、マスター・プロセッサ12及び共同プロセッサ14の間で生データの処理を効果的に分割する。ソナー・アレイ等の遠隔ソース(図1では不図示)は、ポート18を介して、生データに対する先入れ先出し(FIFO)バッファ(不図示)として作用する共有メモリ26の1つの区分に生データをロードする。マスター・プロセッサ12はバス16を介してメモリ26から生データを検索して、マスター・プロセッサ及び共同プロセッサ14はその生データを処理して、バス16を介して必要に応じてデータをそれらの間に転送する。マスター・プロセッサ12はその処理データを共有メモリ26内に規定された別のFIFOバッファ(不図示)にロードし、遠隔ソースがポート20を介してこのFIFOからその処理データを検索する。

## 【0006】

演算例において、計算マシン10は生データに対する $n+1$ 個の各演算を順次実行することによって該生データを処理し、これら演算は一体的に高速フーリエ変換(FFT)等の処理アルゴリズムを構成する。より詳細には、マシン10はマスター・プロセッサ12及び共同プロセッサ14からのデータ処理パイプラインを形成する。クロック信号の所与の周波数で、そうしたパイプラインはしばしばマシン10が単一プロセッサのみを有するマシンよりも高速に生データを処理することを可能としている。

## 【0007】

メモリ26内における生データFIFO(不図示)からの生データ検索後、マスター・プロセッサ12はその生データに対して三角関数等の第1番演算を実行する。この演算は第1番結果を生み出し、それをプロセッサ12がメモリ26内に規定された第1番結果FIFO(不図示)に記憶する。典型的には、プロセッサ12はメモリ22内に記憶されたプログラムを実行し、そのプログラムの制御の下で上述した動作を実行する。プロセッサ12はメモリ22を作業メモリとしても使用し得て、当該プロセッサが第1番演算の中間期間に生成するデータを一時的に記憶する。

## 【0008】

次に、メモリ26内における第1番結果FIFO(不図示)からの第1番結果検索後、共同プロセッサ14はその第1番結果に対して対数関数等の第2番演算を実行する。この第2番演算は第2番結果を生み出し、それを共同プロセッサ14がメモリ26内に規定された第2番結果FIFO(不図示)に記憶する。典型的には、共同プロセッサ14はメモリ24内に記憶されたプログラムを実行し、そのプログラムの制御の下で上述した動作を実行する。共同プロセッサ14はメモリ24を作業メモリとしても使用し得て、当該共同プロセッサが第2番演算の中間期間に生成するデータを一時的に記憶する。

## 【0009】

次に共同プロセッサ24<sub>1</sub>〜24<sub>n</sub>は、共同プロセッサ24<sub>1</sub>に対して先に議論されたものと同様に、(第2番結果〜第 $(n-1)$ 番)結果に対して(第3番演算〜第 $n$ 番)演算を順次実行する。

## 【0010】

共同プロセッサ24<sub>n</sub>によって実行される第 $n$ 番演算は最終結果、即ち処理データを生み出す。共同プロセッサ24<sub>n</sub>はその処理データをメモリ26内に規定された処理データFIFO(不図示)内にロードし、遠隔装置(図1では不図示)がこのFIFOからその処理データを検索する。

## 【0011】

マスター・プロセッサ12及び共同プロセッサ14は処理アルゴリズムの種々の演算を同時に実行するので、計算マシン10は、しばしば、種々の演算を順次実行する単一プロセッサを有する計算マシンよりも生データを高速に処理することができる。詳細には、単一プロセッサは、生データから成る先行集合に対する全 $(n+1)$ 個の演算を実行するまで、生データから成る新しい集合を検索できない。しかし、以上に議論したパイプライン技術を用いて、マスター・プロセッサ12は第1演算だけを実行後に生データから成る新しい集合を検索できる。結果として、所与のクロック周波数でこのパイプライン技術は、単一プロセッサ・マシン(図1では不図示)と比較して約 $n+1$ 倍だけマシン10が生データを処理する速度を増大することができる。

## 【0012】

代替的には、計算マシン10は、生データに対するFFT等の処理アルゴリズムの $(n+1)$ 例を同時に実行することによって該生データを並列して処理し得る。即ち、もしそのアルゴリズムが先行する例において先に記載されたような $(n+1)$ 個の順次演算を含めば、マスター・プロセッサ12及び共同プロセッサ14の各々は生データからそれぞれが成る各集合に対して、順次、全 $(n+1)$ 個の演算を実行する。その結果として、所与のクロック周波数で、先のパイプライン技術と同様のこの並列処理技術は、単一プロセッサ・マシン(図1では不図示)と比較して約 $n+1$ 倍だけマシン10が生データを処理す

る速度を増大することができる。

#### 【0013】

残念ながら、計算マシン10は単一プロセッサ・計算マシン（図1では不図示）と比べてより迅速にデータを処理できるが、マシン10のデータ処理速度はしばしばプロセッサ・クロックの周波数より非常に小さい。詳細には、計算マシン10のデータ処理速度はマスター・プロセッサ12及び共同プロセッサ14がデータ処理するのに必要な時間によって制限される。簡略化のため、この速度制限の例はマスター・プロセッサ12と連携して議論されているが、この議論は共同プロセッサ14にも適用されることを理解して頂きたい。先に議論されたように、マスター・プロセッサ12は所望の方式でデータを操作すべくプロセッサを制御するプログラムを実行する。このプログラムはプロセッサ12が実行する複数の命令から成るシーケンスを含む。残念ながら、プロセッサ12は典型的には単一命令を実行するために多数のクロック・サイクルを必要とし、そしてしばしばデータの単一値を処理すべく多数の命令を実行しなければならない。例えば、プロセッサ12が第1データ値A（不図示）を第2データ値B（不図示）で乗算することを仮定する。第1クロック・サイクル中、プロセッサ12はメモリ22から乗算命令を検索する。第2及び第3クロック・サイクル中、プロセッサ12はメモリ26からA及びBをそれぞれ検索する。第4クロック・サイクル中、プロセッサ12はA及びBを乗算し、そして第5クロック・サイクル中に結果としての積をメモリ22或は26に記憶するか、或は、その結果としての積を遠隔装置（不図示）に提供する。これは最良ケースのシナリオであり、その理由は多くの場合にプロセッサ12はカウンタの初期化及び閉鎖等のオーバーヘッド・タスクに対して付加的なクロック・サイクルを必要とするからである。それ故に、よくてもプロセッサ12はA及びBを処理すべく5クロック・サイクルを必要とするか、或は、1データ値当たり平均2.5クロック・サイクルを必要とする。

#### 【0014】

結果として、計算マシン10がデータを処理する速度は、しばしば、マスター・プロセッサ12及び共同プロセッサ14を駆動するクロックの周波数より非常に低い。例えば、もしプロセッサ12は1.0ギガヘルツ（GHz）でクロックされるが、1データ値当たり平均2.5クロック・サイクルを必要とすれば、効果的なデータ処理速度は（1.0 GHz）／2.5＝0.4 GHzと同等である。この効果的なデータ処理速度は、しばしば、1秒当たり演算数の単位で特徴付けされる。それ故に、この例において、1.0 GHzのクロック速度で、プロセッサ12は0.4ギガ演算数／秒（Gops）で使用限界が定められる。

#### 【0015】

図2は、所与クロック周波数で且つしばしば該パイプラインがクロックされる速度と略同一速度で、プロセッサが可能であるよりは高速で典型的にはデータを処理できるハードウェアに組み込まれたデータ・パイプライン30のブロック図である。パイプライン30は、プログラム命令を実行することなく、各データに対する各演算を各々が実行する演算子回路32<sub>1</sub>～32<sub>n</sub>を含む。即ち、所望の演算は回路32内に「書き込み」が為されて、それがプログラム命令の必要性なしに自動的にその演算を具現化するように為す。プログラム命令の実行と関連されたオーバーヘッドを減ずることによって、パイプライン30は所与のクロック周波数でプロセッサが可能であるよりは単位秒当たりより多くの演算を典型的には実行する。

#### 【0016】

例えば、パイプライン30は所与のクロック周波数でプロセッサが可能であるよりは高速で以下の数式1をしばしば解くことができる。

$$Y(x_k) = (5x_k + 3)2^{x_k}$$

ここで、 $x_k$ は複数の生データ値から成るシーケンスを表す。この例において、演算子回路32<sub>1</sub>は $5x_k$ を計算する乗算器であり、回路32<sub>2</sub>は $5x_k + 3$ を計算する加算器であり、そして回路32<sub>3</sub>（ $n=3$ ）は $(5x_k + 3)2^{x_k}$ を計算する乗算器である。

#### 【0017】

第1クロック・サイクル $k=1$ 中、回路32<sub>1</sub>はデータ値 $x_1$ を受け取って、それを5で乗じて、 $5x_1$ を生成する。

【0018】

第2クロック・サイクル $k=2$ 中、回路32<sub>2</sub>は回路32<sub>1</sub>から $5x_1$ を受け取って、3を加えて、 $5x_1+3$ を生成する。またこの第2クロック・サイクル中に回路32<sub>1</sub>は $5x_2$ を生成する。

【0019】

第3クロック・サイクル $k=3$ 中、回路32<sub>3</sub>は回路32<sub>2</sub>から $5x_1+3$ を受け取って、 $2^{x_1}$ で乗じて（効果としては、 $x_1$ だけ $5x_1+3$ を左シフトする）、第1結果 $(5x_1+3)2^{x_1}$ を生成する。またこの第3クロック・サイクル中に回路32<sub>1</sub>は $5x_3$ を生成し、回路32<sub>2</sub>は $5x_2+3$ を生成する。

10

【0020】

このようにしてパイプライン30は、全ての生データ値が処理されるまで、引き続く生データ値 $x_k$ の処理を続行する。

【0021】

結果として、生データ値 $x_1$ の受け取り後の2つのクロック・サイクルの遅延、即ち、この遅延はパイプライン30の待ち時間としばしば呼称され、パイプラインは結果 $(5x_1+3)2^{x_1}$ を生成し、その後、1つの結果を生成する、即ち各クロック・サイクル毎に $(5x_2+3)2^{x_2}$ 、 $(5x_3+3)2^{x_3}$ 、 $\dots$ 、 $(5x_n+3)2^{x_n}$ を生成する。

【0022】

待ち時間を無視して、パイプライン30はこうしてクロック速度と同等のデータ処理速度を有する。比較して、マスター・プロセッサ12及び共同プロセッサ14（図1）が先の例におけるようにクロック速度の0.4倍であるデータ処理速度を有すると仮定すれば、パイプライン30は、所与のクロック速度で、計算マシン10（図1）よりも2.5倍高速でデータを処理できる。

20

【0023】

更に図2で参照されるように、設計者はフィールド・プログラマブル・ゲート・アレイ（FPGA）等のプログラマブル・ロジックIC（PLIC）にパイプライン30を具現化することを選ぶ可能性があり、その理由はPLICが特殊用途IC（ASIC）が為すよりも多くの設計及び変更の柔軟性を許容するからである。PLIC内にハードウェアに組み込まれた接続を構成するため、設計者はPLIC内に配置された相互接続構成レジスタを単に所定バイナリー状態に設定する。全てのこうしたバイナリー状態の組み合わせはしばしば「ファームウェア」と呼称される。典型的には、設計者はこのファームウェアをPLICと結合された不揮発性メモリ（図2では不図示）内にロードする。PLICを「ターンオン」すると、それはファームウェアをそのメモリから相互接続構成レジスタにダウンロードする。それ故に、PLICの機能を変更すべく、設計者は単にそのファームウェアを変更して、PLICがその変更されたファームウェアを相互接続構成レジスタにダウンロードすることを可能とする。ファームウェアを単に変更することによってPLICを変更する能力は、モデル作成段階中や「フィールド内」にパイプライン30をアップグレードするために特に有用である。

30

【0024】

残念ながら、ハードウェアに組み込まれたパイプライン30は、典型的にはすべてのアルゴリズムを実行できるわけではなく、特に重要な意思決定を引き起こすようなアルゴリズムを実行できない。プロセッサは、典型的には、意思決定命令（例えば、「もしAであれば、Bへ行き、またCへ行く」のように、条件命令）を、比肩する長さの演算命令（例えば、「 $A+B$ 」）を実行できる程に高速に実行できる。しかしパイプライン30は、比較的単純な決定（例えば、「 $A>B?$ 」）を為し得るが、典型的には比較的複雑な決定（例えば、「もしAであれば、Bへ行き、またCへ行く」）を実行することができない。そして、そうした複雑な決定を実行すべくパイプライン30を設計できるが、必要とされる回路のサイズ及び複雑性はしばしばそうした設計を非現実的に為し、特にアルゴリズムが

40

50

多数の種々の複雑な決定を含む場合にそうである。

#### 【0025】

結果として、プロセッサは典型的には重要な意思決定を必要とする用途において使用され、ハードウェアに組み込まれたパイプラインは殆ど意思決定が為されないか或は意思決定されない「ナンバークランチング（数値データ処理）」用途に典型的には限定される。

#### 【0026】

更には、下記に議論されるように、典型的には、特にパイプライン30が多数のPLICを含む場合、図2のパイプライン30等のハードウェアに組み込まれたパイプラインを設計/変更するよりも、図1の計算マシン10等のプロセッサに基づく計算マシンを設計/変更することが非常に易しい。

#### 【0027】

プロセッサ及びそれらの周辺機器（例えば、メモリ）等の計算構成要素は、典型的には、プロセッサに基づく計算マシンを形成すべくそれら構成要素の相互接続を補助する工業規格通信インターフェースを含む。

#### 【0028】

典型的には、規格通信インターフェースは2つの層、即ち、物理層及びサービス層を含む。

#### 【0029】

物理層は、回路とこの回路のインターフェース及び動作パラメータを形成する対応回路相互接続とを含む。例えば、物理層はそれら構成要素を1つのバスに接続するピンと、それらのピンから受け取ったデータをラッチするバッファと、信号をそれらピンに駆動するドライバとを含む。動作パラメータは、ピンが受け取るデータ信号の許容可能電圧範囲と、データの書き込み及び読み取りのための信号タイミングと、動作の支援されたモード（例えば、バーストモード、ページモード）とを含む。従来の物理層はトランジスタートランジスタ論理（TTL）及びRAMBUSを含む。

#### 【0030】

サービス層は、計算構成要素のデータ転送のためのプロトコルを含む。このプロトコルはデータのフォーマットと、構成要素によるフォーマット済みデータの送受信の方式とを含む。従来の通信プロトコルは、ファイル転送プロトコル（FTP）及び伝送制御プロトコル/インターネット・プロトコル（TCP/IP）（拡張）を含む。

#### 【0031】

結果として、製造業者やその他は工業規格通信インターフェースを有する計算構成要素を典型的には設定するので、そうした構成要素のインターフェースを典型的には設計でき、それを他の計算構成要素と比較的少ない労力で相互接続することができる。これは、計算マシンの他の部分の設計に設計者自信の時間を殆ど費やすことを可能として、各種構成要素を追加或は除去することによってそのマシンを変更することを可能としている。

#### 【0032】

工業規格通信インターフェースを支援する計算構成要素を設計することは、設計ライブラリから既存の物理層を用いることによって設計時間を節約することを可能としている。これは、設計者が構成要素を既製の計算構成要素と容易にインターフェースすることを保証するものでもある。

#### 【0033】

そして、共通した工業規格通信インターフェースを支援する計算構成要素を用いる計算マシンを設計することは、設計者がそれら構成要素を少しの時間及び労力で相互接続することを可能としている。それら構成要素は共通インターフェースを支援するので、設計者はそれらをシステム・バスを介して少しの設計労力で相互接続することができる。そして、その支援されたインターフェースは工業規格であるので、マシンを容易に変更することができる。例えば、システム設計が進化するに伴って種々の構成要素及び周辺機器をマシンに追加することができるか、或は、テクノロジーが進化するに伴って次世代の構成要素を追加/設計することが可能である。更には、構成要素が通常の工業規格サービス層を支



援するので、計算マシンのソフトウェアに対応するプロトコルを具現化する既存のソフトウェア・モジュールを組み込むことができる。それ故に、インターフェース設計が本質的には既に整っているもので少しの労力で構成要素をインターフェースでき、よって、マシンに所望の機能を実行させるマシンの各種部分（例えばソフトウェア）の設計に集中することができる。

#### 【0034】

しかし残念ながら、図2のパイプライン30等のハードウェアに組み込まれたパイプラインを形成すべく、使用されるPLIC等の各種構成要素に対する既知の工業規格通信インターフェースが全く知られていない。

#### 【0035】

結果として、多数のPLICを有するパイプラインを設計すべく、「ゼロから」種々のPLICの間の通信インターフェースを設計及びデバッグするのに、多大な時間を費やし且つ多大な労力を行使する。典型的には、そうしたその場限りの通信インターフェースは種々のPLIC間で転送されるデータのパラメータに依存する。同じように、プロセッサとインターフェースするパイプラインを設計すべく、ゼロからのパイプライン及びプロセッサの間の通信インターフェースの設計及びデバッグに関して多大な時間を費やし且つ多大な労力を行使する必要がある。

#### 【0036】

同様に、そうしたパイプラインをPLICを該パイプラインに追加することによって変更すべく、典型的には、その追加されたPLICと既存のPLICとの間の通信インターフェースの設計及びデバッグに関して多大な時間を費やし且つ多大な労力を行使する。同じように、プロセッサを追加することによってパイプラインを変更すべく、或は、パイプラインを追加することによって計算マシンを変更すべく、パイプライン及びプロセッサの間の通信インターフェースの設計及びデバッグに関して多大な時間を費やし且つ多大な労力を行使しなければならないであろう。

#### 【0037】

結果として、図1及び図2で参照されるように、多数のPLICをインターフェースすることとプロセッサをパイプラインにインターフェースすることとの難しさのため、計算マシンを設計する際に多大な妥協を為すことがしばしば強られる。例えば、プロセッサに基づく計算マシンでは、ナンバークランピング速度を、複雑な意思決定を為す能力に対する設計/変更の柔軟性と交換することを強られる。逆に、ハードウェアに組み込まれたパイプラインに基づく計算マシンでは、複雑な意思決定を為す能力と設計/変更の柔軟性を、ナンバークランピング速度と交換することを強られる。更には、多数のPLICをインターフェースすることに関する難しさのため、少数のPLICよりも多くのPLICを有するパイプラインに基づくマシンを設計することはしばしば実際的ではない。その結果、実際的なパイプラインに基づくマシンはしばしば制限された機能しか有しない。そして、プロセッサをPLICとインターフェースすることに関する難しさのため、プロセッサを1つのPLICよりも多くのPLICにインターフェースすることは実際的ではない。その結果、プロセッサ及びパイプラインを組み合わせることによって獲得される利益は最少となる。

#### 【発明の開示】

#### 【発明が解決しようとする課題】

#### 【0038】

それ故に、プロセッサに基づくマシンの意思決定を為す能力を、ハードウェアに組み込まれたパイプラインに基づくマシンのナンバークランピング速度と組み合わせることを可能とする新しい計算アーキテクチャに対する要望が生じてきている。

#### 【課題を解決するための手段】

#### 【0039】

本発明の実施例において、計算マシンは、第1バッファと該バッファと結合されたプロセッサを含む。このプロセッサは、アプリケーション、第1データ転送オブジェクト、

並びに、第2データ転送オブジェクトを実行し、そのアプリケーションの制御下でデータを発行し、第1データ転送オブジェクトの制御下でその発行されたデータをバッファにロードしてから、第2データ転送オブジェクトの制御下でそのバッファから発行されたデータを検索するように動作できる。

#### 【0040】

本発明の別の実施例に従えば、プロセッサは、データを検索し、第1データ転送オブジェクトの制御下でその検索されたデータをバッファにロードし、第2データ転送オブジェクトの制御下でそのバッファからデータをアンロードしてから、アプリケーションの制御下でそのアンロードされたデータを処理する。

#### 【0041】

計算マシンがプロセッサと結合されたハードウェアに組み込まれたパイプライン加速器を含むビアーベクトル・マシンである場合、バッファ及びデータ転送オブジェクトはデータの（単方向性であろうが二方向性であろうが）転送を補助する。

#### 【発明を実施するための最良の形態】

#### 【0042】

図3は、本発明の一実施例に従ったビアーベクトル・アーキテクチャを有する計算マシン40の概略ブロック線図である。ホストプロセッサ42に加えて、ビアーベクトル・マシン40はパイプライン加速器44を含み、それがデータ処理の少なくとも一部を実行して、図1の計算マシン10における共同プロセッサ14の列と効果的に置き換わる。それ故に、ホストプロセッサ42及び加速器44はデータ・ベクトルを前後に転送できる「ピア」である。加速器44はプログラム命令を実行しないので、所与のクロック周波数で共同プロセッサの列ができるものよりも著しく高速にデータに対して数学的に集中的な演算を典型的には実行する。結果として、プロセッサ42の意思決定能力と加速器44のナンバークラッシュ能力とを組み合わせることによって、マシン40はマシン10等の従来の計算マシンと同一の能力を有するが、しばしばそれよりもデータをより高速に処理することができる。更には、以下に議論されると共に、先行して引用された「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された特許文献3に議論されているように、加速器44にホストプロセッサ42と同一の通信インターフェースを設けることが、特にその通信インターフェースが工業規格である場合に、マシン40の設計及び変更を補助する。そして、加速器44が多数の構成要素（例えば、複数のPLIC）を含む場合、それら構成要素に同一の通信インターフェースを設けることは、特にその通信インターフェースが工業規格である場合に、加速器の設計及び変更を補助する。更には、マシン40は以下に議論されると共に先行して引用された特許出願におけるような他の長所等をも提供し得る。

#### 【0043】

更に図3で参照されるように、ホストプロセッサ42及びパイプライン加速器44に加えて、ビアーベクトル計算マシン40は、プロセッサ・メモリ46、インターフェース・メモリ48、バス50、ファームウェア・メモリ52、任意選択的な生データ入力ポート54、56、処理済みデータ出力ポート58、60、並びに、任意選択的なルータ61を含む。

#### 【0044】

ホストプロセッサ42は処理ユニット62及びメッセージ・ハンドラー64を含み、プロセッサ・メモリ46は処理ユニット・メモリ66及びハンドラー・メモリ68を含み、そのそれぞれがプロセッサ・ユニット及びメッセージ・ハンドラーに対するプログラム及び作業の両メモリとして役立っている。プロセッサ・メモリ46は、加速器コンフィギュレーション・レジストリ70及びメッセージ・コンフィギュレーション・レジストリ72をも含み、それらが、ホストプロセッサ42に加速器44の機能を構成させると共に、メッセージ・ハンドラー64が送信及び受信するメッセージの構造を構成させることを可能とするそれぞれのコンフィギュレーション・データを記憶する。

#### 【0045】

パイプライン加速器 44 は少なくとも 1 つの P L I C (不図示) 上に配置され、プログラム命令を実行することなしに各データを処理するハードウェアに組み込まれたパイプライン 74<sub>1</sub> - 74<sub>n</sub> を含む。ファームウェア・メモリ 52 は加速器 44 に対するコンフィギュレーション・ファームウェアを記憶する。もし加速器 44 が多数の P L I C 上に配置されたら、それら P L I C 及びそれらの各ファームウェア・メモリは多数の回路ボード上、即ちドーターカード (不図示) 上に配置され得る。加速器 44 及びドーターカードは、先行して引用された「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された特許文献 3 と、「多数パイプライン・ユニットを有するパイプライン加速器、関連計算マシン、並びに、方法」と題された特許文献 5 に更に議論されている。代替的には、加速器 44 は、少なくとも 1 つの A S I C 上に配置され得て、よって構成不可能である内部相互接続を有し得る。この代替例において、マシン 40 はファームウェア・メモリ 52 を省略し得る。更には、加速器 44 が多数パイプライン 74 を含んで示されているが、ただ 1 つのパイプラインを含み得る。加えて、図示されていないが、加速器 44 はデジタル信号プロセッサ (D S P) 等の 1 つ或はそれ以上のプロセッサを含み得る。

#### 【0046】

ビームベクトル・マシン 40 の一般動作は、先行して引用された「改善された計算アーキテクチャ、関連システム、並びに、方法」と題された特許文献 2 に議論されており、ホストプロセッサ 42 の機能ブロック図及び動作は図 4 乃至図 7 と連携して以下に議論されている。図 4 は、本発明の実施例に従った図 3 のホストプロセッサ 42 及びパイプライン・バス 50 の機能ブロック線図である。一般に、処理ユニット 62 は 1 つ或はそれ以上のソフトウェア・アプリケーションを実行し、メッセージ・ハンドラー 64 は、ソフトウェア・アプリケーション (単数或は複数) とパイプライン加速器 44 (図 3) の間でデータを転送する 1 つ或はそれ以上のソフトウェア・オブジェクトを実行する。データ処理、データ転送、並びに、他の機能を種々のアプリケーション及びオブジェクトの間で分割することは、ホストプロセッサ・ソフトウェアのより容易な設計及び変更をもたらす。更には、以下の記載においてソフトウェア・アプリケーションが特定の動作を実行するように説明されているが、処理ユニット 62 或はメッセージ・ハンドラー 64 は、実際の動作において、ソフトウェア・アプリケーションを実行してそのアプリケーションの制御下でその動作を実行することを理解して頂きたい。同じように、以下の記載においてソフトウェア・オブジェクトが特定の動作を実行するように説明されているが、処理ユニット 62 或はメッセージ・ハンドラー 64 は、実際の動作において、ソフトウェア・オブジェクトを実行してそのオブジェクトの制御下でその動作を実行することを理解して頂きたい。

#### 【0047】

更に図 4 で参照されるように、処理ユニット 62 は、データ処理アプリケーション 80、加速器例外マネージャ・アプリケーション (以降、例外マネージャ) 82、並びに、加速器コンフィギュレーション・マネージャ・アプリケーション (以降、コンフィギュレーション・マネージャ) 84 を実行するが、それらアプリケーションは集合的に処理ユニット・アプリケーションと呼称される。データ処理アプリケーションは、パイプライン加速器 44 (図 3) と共同してデータを処理する。例えば、データ処理アプリケーション 80 はポート 54 (図 3) を介して生センサー・データを受信し、そのデータをパースし、そのパースされたデータを加速器 44 に送信し得て、そして加速器はそのパースされたデータに F F T を実行して、その処理されたデータを更なる処理のためにデータ処理アプリケーションに戻し得る。例外マネージャ 82 は加速器 44 からの例外メッセージを取り扱い、コンフィギュレーション・マネージャ 84 は加速器のコンフィギュレーション・ファームウェアをビームベクトル・マシン 40 (図 3) の初期化中にメモリ 52 にロードする。またコンフィギュレーション・マネージャ 84 は、例えば加速器の起動動作に応じて、初期化後に加速器 44 を再構成し得る。図 6 及び図 7 と連携されて以下に更に議論されるように、処理ユニット・アプリケーションは、破線 85、87、89 で示されるように、直接的に相互に通信し得るか、或は、データ転送オブジェクト 86 を介して相互に通信し得る。

メッセージ・ハンドラー 64 はデータ転送オブジェクト 86、通信オブジェクト 88、並びに、入力及び出力読み取りオブジェクト 90、92 を実行して、入力及び出力キュー・オブジェクト 94、96 を実行し得る。データ転送オブジェクト 86 は、通信オブジェクト 88 及び処理ユニット・アプリケーションの間でデータを転送し、インターフェース・メモリ 48 をデータ・バッファとして使用し得て、処理ユニット・アプリケーション及び加速器 44 にそれぞれ独立して動作させることを可能としている。例えば、メモリ 48 は、しばしばデータ処理アプリケーション 80 よりも高速である加速器 44 に、そのデータ処理アプリケーションを「待機」することなしに動作させることを可能としている。通信オブジェクト 88 はデータ・オブジェクト 86 及びパイプライン・バス 50 の間でデータを転送する。入力及び出力読み取りオブジェクト 90、92 は、データ転送オブジェクト 10  
が通信オブジェクト 88 及び処理ユニット・アプリケーションの間でデータを転送している際にそれらのデータ転送オブジェクト 86 を制御する。そして、実行されると、入力及び出力キュー・オブジェクト 94、96 は入力及び出力読み取りオブジェクト 90、92 に所望の優先順位に従ってデータのその転送を同期させている。

#### 【0048】

更には、ビアーベクトル・マシン 40 (図 3) の初期化中、メッセージ・ハンドラー 64 は、メッセージ・コンフィギュレーション・レジストリ 72 (図 3) に記憶されたコンフィギュレーション・データからデータ転送オブジェクト 86 を例示する従来のオブジェクト・ファクトリー 98 を例示し実行する。このメッセージ・ハンドラー 64 も、メッセージ・コンフィギュレーション・レジストリ 72 に記憶されたコンフィギュレーション・データから通信オブジェクト 88、入力及び出力リーダー・オブジェクト 90、92、並びに、入力及び出力キュー・オブジェクト 94、96 を例示する。結果として、レジストリ 72 に記憶されたコンフィギュレーション・データを単に設計及び変更することによって、それらソフトウェア・オブジェクトそしてそれ故、それらのデータ転送パラメータを設計及び変更し得る。典型的にはこれは、ソフトウェア・オブジェクト各々を個々別々に設計或は変更するよりもより少ない時間消費で済む。

#### 【0049】

図 4 のホストプロセッサ 42 の動作は図 5 乃至図 7 と連携して以下に議論される。

#### 【0050】

<データ処理>

図 5 は、本発明の実施例に従った図 4 のデータ処理アプリケーション 80、データ転送オブジェクト 86、並びに、インターフェース・メモリ 48 の機能ブロック線図である。

#### 【0051】

データ処理アプリケーション 80 は、各データ処理動作をそれぞれ実行する多数のスレッド 100<sub>1</sub>~100<sub>n</sub>を含む。例えば、スレッド 100<sub>1</sub> は加算を、スレッド 100<sub>2</sub> は減算をそれぞれ実行し得るか、或は、スレッド 100<sub>1</sub> 及び 100<sub>2</sub> の双方は加算を実行し得る。

#### 【0052】

各スレッド 100 は、パイプライン加速器 44 (図 3) に仕向けられたデータを生成、即ち発行して、加速器からデータを受信、即ち加入するか、或は、データの発行及び加入の双方を為す。例えば、スレッド 100<sub>1</sub>~100<sub>n</sub> の各々は加速器 44 からデータの発行及び加入の双方を為す。スレッド 100 は別のスレッド 100 と直に通信も為し得る。例えば、破線 102 で示されるように、スレッド 100<sub>3</sub> 及び 100<sub>4</sub> は直接相互に通信し得る。更にはスレッド 100 は加速器 44 (図 3) 以外の構成要素 (不図示) からデータを受信するか或は該構成要素にデータを送信し得る。しかし、簡潔さのため、スレッド 100 とそうした別の構成要素の間でのデータ転送の議論は省略される。

#### 【0053】

更に図 5 で参照されるように、インターフェース・メモリ 48 及びデータ転送オブジェクト 86<sub>1</sub>~86<sub>n</sub> は、各スレッド 100 及び通信オブジェクト 88 の間でデータを転送するため、多数の単向性チャネル 104<sub>1</sub>~104<sub>n</sub> を機能的に形成する。インターフェー

ス・メモリ 48 は、単位チャネル 104 当たり 1 つのバッファとなるように、多数のバッファ 106<sub>1</sub>、106<sub>2</sub>を含む。これらバッファ 106 は、各々、データの単一グループ分け（例えば、バイト、ワード、ブロック）、或は、バッファの少なくとも幾分かはそれぞれがデータから成る各多数グループ分けを記憶できる F I F O バッファであり得る。各チャネル 104 には 2 つのデータ・オブジェクト 86 があり、一方が各スレッド 100 及び各バッファ 106 の間でデータを転送するものであり、他方がバッファ 106 及び通信オブジェクト 88 の間でデータを転送するものである。例えば、チャネル 104<sub>1</sub> は、バッファ 106<sub>1</sub>、発行されたデータをスレッド 100<sub>1</sub> からバッファ 106<sub>2</sub> に転送するデータ転送オブジェクト 86<sub>1</sub>、並びに、発行されたデータをバッファ 106<sub>1</sub> から通信オブジェクト 88 に転送するデータ転送オブジェクト 86<sub>2</sub>を含む。各々の許容できるデータ転送に対して各チャネル 104 を含むことは、データ・ボトルネックに対するポテンシャルを低減すると共に、ホストプロセッサ 42（図 4）の設計及び変更を補助する。

10

#### 【0054】

図 3 乃至図 5 で参照されるように、初期化中及びデータ処理アプリケーション 80、データ転送オブジェクト 86、通信オブジェクト 88、並びに、任意選択的なリーダ及びキュー・オブジェクト 90、92、94、96 を実行している最中におけるホストプロセッサ 42 の動作は、本発明の実施例に従って議論される。

#### 【0055】

ホストプロセッサ 42 の初期化中、オブジェクト・ファクトリー 98 はデータ転送オブジェクト 86 を例示しバッファ 104 を規定する。詳細には、オブジェクト・ファクトリー 98 はレジストリ 72 からコンフィギュレーション・データをダウンロードし、データ処理アプリケーション 80 が必要とし得る各データ転送オブジェクト 86<sub>1</sub>に対するソフトウェア・コードを生成する。アプリケーション 80 が必要とし得るデータ転送オブジェクトの識別性はコンフィギュレーション・データの典型的には一部であるが、アプリケーション 80 はデータ転送オブジェクト 86 の全てを使用する必要はない。次いで、生成されたオブジェクト 86<sub>1</sub>から、オブジェクト・ファクトリー 98 はデータ・オブジェクト 86<sub>2</sub>をそれぞれ例示する。典型的には、以下の例で議論されるように、オブジェクト・ファクトリー 98 は、同一ソフトウェア・コードの多数例証として同一のバッファ 104 にアクセスするデータ転送オブジェクト 86<sub>1</sub>及び 86<sub>2</sub>を例示する。これは、さもなければオブジェクト・ファクトリー 98 が約半分で生成することになるコード量を低減する。更には、メッセージ・ハンドラー 64 は、もしあれば、アプリケーション 80 が必要としないデータ転送オブジェクト 86 を決定し得ると共に、それら不必要なデータ転送オブジェクトの例証を削除し得てメモリを節約する。代替的には、メッセージ・ハンドラー 64 はオブジェクト・ファクトリー 98 がデータ転送オブジェクト 86 を生成する前にこの決定を為し得て、オブジェクト・ファクトリーにアプリケーション 80 が必要とするデータ転送オブジェクトだけを例示させる。加えて、データ転送オブジェクト 86 が、各バッファ 104 が位置決めされているインターフェース・メモリ 48 のアドレスを含むので、オブジェクト・ファクトリー 98 はデータ転送オブジェクトを例示する際にバッファのサイズ及び箇所を効果的に規定する。

20

30

#### 【0056】

例えば、オブジェクト・ファクトリー 98 は以下の方式でデータ転送オブジェクト 86<sub>1</sub>及び 86<sub>2</sub>を例示する。先ず、ファクトリー 98 はレジストリ 72 からコンフィギュレーション・データをダウンロードし、データ転送オブジェクト 86<sub>1</sub>及び 86<sub>2</sub>に対する共通ソフトウェア・コードを生成する。次に、ファクトリー 98 はその共通ソフトウェア・コードの各例証としてデータ転送オブジェクト 86<sub>1</sub>及び 86<sub>2</sub>を例示する。即ち、メッセージ・ハンドラー 64 は共通ソフトウェア・コードをハンドラー・メモリ 68 の 2 つの箇所或は他のプログラム・メモリ（不図示）に効果的にコピーし、一方の箇所をオブジェクト 86<sub>1</sub>として実行し、他方の箇所をオブジェクト 86<sub>2</sub>として実行する。

40

#### 【0057】

更に図 3 乃至図 5 で参照されるように、ホストプロセッサ 42 の初期化後、データ処理

50

アプリケーション 80 はデータを処理し、パイプライン加速器 44 にデータを送信し該パイプライン加速器からデータを受信する。

【0058】

加速器 44 にデータを送信するデータ処理アプリケーション 80 の例は、チャンネル 104<sub>1</sub>と連携して議論される。

【0059】

先ず、スレッド 100<sub>1</sub> はデータを生成して、データ転送オブジェクト 86<sub>1a</sub> にデータを発行する。スレッド 100<sub>1</sub> は、加速器 44（以下に更に議論される）から或はポート 54 を介してソナー・アレイ或はデータバス等の別のソース（不図示）から受信する生データに対して演算することによってデータを生成し得る。

【0060】

次いで、データ・オブジェクト 86<sub>1a</sub> は発行されたデータをバッファ 106<sub>1</sub> にロードする。

【0061】

次にデータ転送オブジェクト 86<sub>1a</sub> は、バッファ 106<sub>1</sub> にデータ転送オブジェクト 86<sub>1a</sub> から新しく発行されたデータがロードされたかを決定する。出力リーダー・オブジェクト 92 は、バッファ 106<sub>1</sub> を新しく発行されたデータに対して照合させるようにデータ転送オブジェクト 86<sub>1a</sub> に周期的に命令する。代替的には、出力リーダー・オブジェクト 92 は、バッファ 106<sub>1</sub> が新しく発行されたデータをいつ受信したかをデータ転送オブジェクト 86<sub>1a</sub> に通知する。詳細には、出力キュー・オブジェクト 96 は発行されたデータをバッファ 106<sub>1</sub> に記憶するデータ転送オブジェクト 86<sub>1a</sub> に応じて、固有識別子（不図示）を生成し記憶する。この識別子に応じて、出力リーダー・オブジェクト 92 は、バッファ 106<sub>1</sub> が新しく発行されたデータを含むことをデータ転送オブジェクト 86<sub>1a</sub> に通知する。多数のバッファ 106 が各新しく発行されたデータを含む場合、出力キュー・オブジェクト 96 はそのデータが発行される順番を記録し得て、出力リーダー・オブジェクト 92 は各データ転送オブジェクト 86<sub>1a</sub> に同一の順番で通知し得る。よって、出力リーダー・オブジェクト 92 及び出力キュー・オブジェクト 96 は、各データ転送オブジェクト 86<sub>1a</sub> が加速器 44 に送信する第 1 データとなるように第 1 データを発行させ、各データ転送オブジェクト 86<sub>1a</sub> が加速器 44 に送信する第 2 データとなるように第 2 データを発行させ等々することによってデータ転送を同期する。多数のバッファ 106 が各新しく発行されたデータを含む場合の別の代替例において、出力リーダー及び出力キュー・オブジェクト 92 及び 96 は、その先入れ先出し方式以外の、或は、その先入れ先出し方式に加えて、優先順位方式を具現化し得る。例えば、スレッド 100<sub>1</sub> が第 1 データを発行し、続いてスレッド 100<sub>2</sub> が第 2 データを発行するが、出力キュー・オブジェクト 96 にその第 2 データと関連された優先順位フラグも発行する。第 2 データが第 1 データを凌ぐ優先順位を有するので、出力リーダー・オブジェクト 92 はデータ転送オブジェクト 86<sub>1a</sub> に、データ転送オブジェクト 86<sub>1a</sub> にバッファ 106<sub>1</sub> における発行された第 1 データを通知する前に、バッファ 106<sub>2</sub> における発行された第 2 データを通知する。

【0062】

次いで、データ転送オブジェクト 86<sub>1a</sub> はバッファ 106<sub>1</sub> からその発行データを検索し、そのデータを所定の方式でフォーマットする。例えば、オブジェクト 86<sub>1a</sub> は発行データ（即ち、ペイロード）と、例えば加速器 44 内のデータの仕向先を識別する、ヘッダとを含むメッセージを生成する。このメッセージは高速 I/O（入力／出力）フォーマット等の工業規格フォーマットを有し得る。そうしたメッセージの生成は従来技術であるので、更に議論されない。

【0063】

データ転送オブジェクト 86<sub>1a</sub> による発行データのフォーマット後、それはそのフォーマットされたデータを通信オブジェクト 88 に送信する。

【0064】

次に、通信オブジェクト 88 はフォーマットされたデータをバス 50 を介してパイプ

イン加速器 44 に送信する。通信オブジェクト 88 は、ホストプロセッサ 42 及び加速器 44 の間でデータを転送すべく使用される通信プロトコル（例えば、高速 I/O、TCP/IP）を具現化するように設計されている。例えば、通信オブジェクト 88 はプロトコルが必要とする必須のハンドシェーキング及び他の転送パラメータ（例えば、バス 50 上のメッセージの送受信を仲裁する）を具現化する。代替的には、データ転送オブジェクト 86<sub>2b</sub> は通信プロトコルを具現化し得て、通信オブジェクト 88 は省略され得る。しかしながら、この後者の代替例はデータ転送オブジェクト 86<sub>2b</sub> 全てに追加のコード及び機能を含むように要求するので効率性を欠いている。

#### 【0065】

パイプライン加速器 44 は、次いで、フォーマットされたデータを受信し、メッセージからそのデータを回復し（例えば、ヘッダーが存在する場合にそのデータをヘッダーから分離する）、そのデータを加速器内の適切な仕向先に向かわせ、そのデータを処理する。

#### 【0066】

更に図 3 乃至図 5 で参照されるように、データをホストプロセッサ 42（図 3）に送信するパイプライン加速器 44（図 3）の例はチャンネル 104<sub>2</sub> と連携して議論される。

#### 【0067】

先ず、パイプライン加速器 44 はデータを生成してフォーマットする。例えば、加速器 44 はデータ・パイロードと、例えばそのデータを受信して処理することとなるスレッドである仕向先スレッド 100<sub>1</sub> 及び 100<sub>2</sub> を識別するヘッダーとを含むメッセージを生成する。先に議論されたように、このメッセージは高速 I/O（入力/出力）フォーマット等の工業規格フォーマットを有し得る。

#### 【0068】

次に、加速器 44 はそのフォーマットされたデータを従来方式でバス 50 に駆動する。

#### 【0069】

次いで、通信オブジェクト 88 はバス 50 からそのフォーマットされたデータを受信し、そのフォーマットされたデータをデータ転送オブジェクト 86<sub>2b</sub> に提供する。一実施例において、そのフォーマットされたデータはメッセージの形態であり、通信オブジェクト 88 はそのメッセージ・ヘッダー（先に議論されたように、仕向先スレッド 100<sub>1</sub> 及び 100<sub>2</sub> を識別する）を分析し、そのメッセージをヘッダーに応じてデータ転送オブジェクト 86<sub>2b</sub> に提供する。別の実施例において、通信オブジェクト 88 はメッセージをデータ転送オブジェクト 86<sub>2b</sub> の全てに提供し、それら各々はそのメッセージ・ヘッダーを分析し、もしその機能がデータを仕向先スレッド 100<sub>1</sub> 及び 100<sub>2</sub> に提供することであるときのみ、メッセージを処理する。結果として、この例において、データ転送オブジェクト 86<sub>2b</sub> だけがそのメッセージを処理する。

#### 【0070】

次に、データ転送オブジェクト 86<sub>2b</sub> は通信オブジェクト 88 から受信されたデータをバッファ 106<sub>2</sub> にロードする。例えば、もしそのデータがメッセージ・パイロード内に含まれていれば、データ転送オブジェクト 86<sub>2b</sub> はそのメッセージからデータを回復し（例えば、ヘッダーを分割することによって）、その回復されたデータをバッファ 106<sub>2</sub> にロードする。

#### 【0071】

次いで、データ転送オブジェクト 86<sub>2b</sub> は、バッファ 106<sub>2</sub> がデータ転送オブジェクト 86<sub>2b</sub> から新しいデータを受信したことを決定する。入力リーダー・オブジェクト 90 はデータ転送オブジェクト 86<sub>2b</sub> に、新しく受信されたデータに対してバッファ 106<sub>2</sub> を照合するように周期的に命令し得る。代替的には、入力リーダー・オブジェクト 90 はデータ転送オブジェクト 86<sub>2b</sub> に、バッファ 106<sub>2</sub> が新しく発行されたデータをいつ受信したかを通知する。詳細には、入力キュー・オブジェクト 94 は、その発行されたデータをバッファ 106<sub>2</sub> に記憶するデータ転送オブジェクト 86<sub>2b</sub> に応じて、固有識別子（不図示）を生成し記憶する。この識別子に応じて、入力リーダー・オブジェクト 90 はデータ転送オブジェクト 86<sub>2b</sub> にバッファ 106<sub>2</sub> が新しく発行されたデータを含むことを通知す

る。出力リーダー及び出力キュー・オブジェクト 92 及び 96 と連携して先に議論されたように、多数のバッファ 106 が各新しく発行されたデータを含む場合、入力キュー・オブジェクト 94 は、このデータが発行された順番を記録し得て、入力リーダー・オブジェクト 90 は各データ転送オブジェクト 86<sub>jk</sub>に同一の順番で通知し得る。代替的には、多数のバッファ 106 が各新しく発行されたデータを含む場合、入力リーダー及び入力キュー・オブジェクト 90 及び 94 は、この先入れ先出し方式以外の、或は、この先入れ先出し方式に加えて優先順位方式を具現化し得る。

#### 【0072】

次に、データ・オブジェクト 86<sub>2a</sub>はバッファ 106<sub>2</sub>から加入者スレッド 100<sub>1</sub>及び 100<sub>2</sub>にデータを転送し、それら加入者スレッドはデータに対して各演算を実行する。 10

#### 【0073】

図 5 で参照されるように、他方のスレッドからデータを受信して処理する一方のスレッドの例はスレッド 100<sub>3</sub>によって発行されたデータを受信して処理するスレッド 100<sub>4</sub>と連携して議論される。

#### 【0074】

一実施例において、スレッド 100<sub>3</sub>は任意選択的な接続（破線）102を介してスレッド 100<sub>4</sub>に直接的にデータを発行する。

#### 【0075】

別の実施例において、スレッド 100<sub>3</sub>はチャネル 104<sub>3</sub>及び 104<sub>4</sub>を介してスレッド 100<sub>4</sub>にデータを発行する。詳細には、データ転送オブジェクト 86<sub>3a</sub>は発行されたデータをバッファ 106<sub>3</sub>にロードする。次にデータ転送オブジェクト 86<sub>3b</sub>はバッファ 106<sub>3</sub>からデータを検索し、そのデータを通信オブジェクト 88 に転送し、該通信オブジェクトはデータ転送オブジェクト 86<sub>4a</sub>にデータを発行する。次いでデータ転送オブジェクト 86<sub>4b</sub>はバッファ 106<sub>4</sub>にデータをロードする。次にデータ転送オブジェクト 86<sub>4c</sub>はバッファ 106<sub>4</sub>からスレッド 100<sub>4</sub>にデータを転送する。代替的には、データがバス 50 を介して転送されていないので、データ転送オブジェクト 86<sub>3a</sub>がデータをバッファ 106<sub>3</sub>に直接的にロードし、よって通信オブジェクト 88 及びデータ転送オブジェクト 86<sub>4a</sub>を迂回するようにそのデータ転送オブジェクトを変更し得る。しかし、データ転送オブジェクト 86<sub>3a</sub>を他のデータ転送オブジェクト 86 とは異なるように変更することはメッセージ・ハンドラー 64 の複雑性モジュール方式を増大し得る。 20 30

#### 【0076】

更に図 5 で参照されるように、追加のデータ転送技術が想定されている。例えば、単一スレッドは各多数チャネルを介してパイプライン加速器 44（図 3）内の多数箇所にデータを発行し得る。代替的には、先行して引用された「改善された計算アーキテクチャ、関連システム、並びに、方法」と題された特許文献 2 や、「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された特許文献 3 に議論されているように、加速器 44 は単一チャネル 104 を介してデータを受信し、それを加速器内の多数箇所に提供し得る。更には多数のスレッド（例えば、スレッド 100<sub>1</sub>及び 100<sub>2</sub>）は同一チャネル（例えばチャネル 104<sub>2</sub>）からのデータに加入し得る。加えて、多数のスレッド（例えば、スレッド 100<sub>2</sub>及び 100<sub>3</sub>）は同一チャネル（例えば、チャネル 104<sub>3</sub>）を介して加速器 44 内の同一箇所にデータを発行し得るが、それらスレッドは各チャネル 104 を介して同一の加速器箇所にデータを発行し得る。 40

#### 【0077】

図 6 は、本発明の実施例に従った例外マネージャ 82、データ転送オブジェクト 86、並びに、インターフェース・メモリ 48 の機能的ブロック線図である。

#### 【0078】

例外マネージャ 82 は、パイプライン加速器 44（図 3）の初期化中或は動作中に生じ得る例外を受信しログする。一般に例外は、加速器 44 が所望されない方式で行動する設計者規定事象である。例えば、オーバーフローしているバッファ（不図示）は 1 つの例外であり得て、よって加速器 44 に例外メッセージを生成させて、それを例外マネージャ 8 50



2に送信させる。例外メッセージの生成は、先行して引用された「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された特許文献3に議論されている。

【0079】

例外マネージャ82は、加速器44（図3）の初期化中或は動作中に生ずる例外を取り扱うこともできる。例えば、もし加速器44がオーバーフローしているバッファ（不図示）を含めば、例外マネージャ82は加速器にバッファのサイズを増大させて、将来のオーバーフローを防止させ得る。或は、もし加速器44の1つの区分が誤動作すれば、例外マネージャ82は加速器の別の区分に或はデータ処理アプリケーション80に、誤動作区分に実行が意図された動作を実行させる。そうした例外取り扱いは以下に更に議論されると共に、先行して引用された「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに、方法」と題された特許文献3に更に議論されている。

10

【0080】

加速器例外をログ及び／或は取り扱うべく、例外マネージャ82は1つ或はそれ以上の加入者スレッド100（図5）からのデータに加入し、そのデータから例外が生じたかを決定する。

【0081】

一代替例において、例外マネージャ82は加入者スレッド100（図5）が加入するものと同じのデータに加入する。詳細には、マネージャ82は同一の各チャンネル104<sub>i</sub>（例えば図5のチャンネル104<sub>2</sub>を含む）を介してそのデータを受信し、そのチャンネルから受信する加入者スレッド100（例えば、図5のスレッド100<sub>1</sub>及び100<sub>2</sub>を含む）がそのデータを受信する。結果として、チャンネル104<sub>i</sub>はそのデータを、それらチャンネルがそのデータを加入者スレッド100に提供するのと同じ方式で例外マネージャ82にそのデータを提供する。

20

【0082】

別の代替例において、例外マネージャ82は専用チャンネル106（不図示）からのデータに加入し、該チャンネルが加入者チャンネル104<sub>i</sub>を介してスレッド100にデータを提供しない加速器44（図3）の区分からデータを受信し得る。そうした専用チャンネル104が使用される場合、オブジェクト・ファクトリー98（図4）は、図4と連携して先に議論されたように、ホストプロセッサ42の初期化中、それらチャンネルに対してデータ転送オブジェクト86を生成する。例外マネージャ82は、独占排他的に或は加入者チャンネル104<sub>i</sub>に加えて、専用チャンネル106に加入し得る。

30

【0083】

例外が生じたかを決定すべく、例外マネージャ82はそのデータをメモリ66（図3）内におけるレジストリ（不図示）に記憶された例外コードと比較する。もしそのデータがそれらコードの内の1つと符合すれば、例外マネージャ82は符合されたコードと対応しているその例外が生じたことを決定する。

【0084】

別の代替例において、例外マネージャ82はデータを分析して、例外が生じたかを決定する。例えば、そのデータは加速器44によって実行された動作の結果を表し得る。例外マネージャ82は、そのデータがエラーを含むかを決定し、もしそうであれば、例外が生じたことを決定すると共にその例外の識別性を決定する。

40

【0085】

例外が生じたことの決定後、例外マネージャ82は、例えば対応する例外コードと発現の時間とをログし、加速器44のデバッグ時等の後の使用に備える。例外マネージャ82は、その例外の識別性を決定して、それを例えばシステム設計者に従来方式で伝える。

【0086】

代替的には、例外のログに加えて、例外マネージャ82はその例外を取り扱うための適切な手続きを具現化し得る。例えば、例外マネージャ82は例外取り扱い命令を加速器44、データ処理アプリケーション80、或は、コンフィギュレーション・マネージャ84

50

に送信することによってその例外を取り扱うことができる。例外マネージャ 82 は、発行  
者スレッド 100 (例えば、図 5 のスレッド 100<sub>i</sub>) がデータを発行する媒介となる同  
一の各チャンネル 104<sub>j</sub> (例えば、図 5 のチャンネル 104<sub>j</sub>) を介してか、或は、図 5 と連  
携して先に議論されたように動作する専用例外取り扱いチャンネル 104 (不図示) を通じ  
てかの何れかで、加速器 44 に例外取り扱い命令を送信し得る。もし例外マネージャ 82  
が他のチャンネル 104 を介して命令を送信すれば、オブジェクト・ファクトリー 98 (図  
4) は、図 4 と連携して先に議論されたように、ホストプロセッサ 42 の初期化中にそれ  
らチャンネルに対するデータ転送オブジェクト 86 を生成する。例外マネージャ 82 は、デ  
ータ処理アプリケーション 80 及びコンフィギュレーション・マネージャ 84 に例外取  
扱い命令を直接的 (図 4 における破線 85 及び 89 で示されるように) か、或は、オブ  
ジェクト・ファクトリー 98 がホストプロセッサ 42 の初期化中に更に生成するチャンネル 1  
04<sub>dup1</sub>, 104<sub>dup2</sub> (アプリケーション 80) 及びチャンネル 104<sub>cs1</sub>, 104<sub>cs2</sub> (コ  
ンフィギュレーション・マネージャ 84) を介してかの何れかで発行し得る。

10

#### 【0087】

更に図 6 で参照されるように例外取り扱い命令は、以下に議論されるように、加速器 4  
4、データ処理アプリケーション 80、或は、コンフィギュレーション・マネージャ 84  
に様々な方法で対応する例外を取り扱わせ得る。

#### 【0088】

加速器 44 に送信されると、例外取り扱い命令はその加速器のソフト・コンフィギュ  
レーション或は機能を変更し得る。例えば、先に議論したように、もし例外がバッファ・オ  
ーバフローであれば、その命令は加速器のソフト・コンフィギュレーションを変更し得  
て (即ち、ソフト・コンフィギュレーション・レジスタの内容を変更することによって)  
、バッファのサイズを増大する。或は、もし特定の動作を実行する加速器 44 の区分が誤  
動作していれば、その命令は加速器にディスエーブルされた区分を「オフライン」にさせ  
ることによって加速器の機能を変更し得る。この後者の場合、例外マネージャ 82 は、追  
加命令を介して、加速器 44 の別の区分に、或は、データ処理アプリケーション 80 に、  
以下に議論されるようにディスエーブルされた加速器の区分から動作の「引き継ぎ」を為  
させ得る。加速器 44 のソフト・コンフィギュレーションを改変することは、先行して引  
用された「改善された計算アーキテクチャ用パイプライン加速器、関連システム、並びに  
、方法」と題された特許文献 3 に更に議論されている。

20

30

#### 【0089】

データ処理アプリケーション 80 に送信されると、例外取り扱い命令はそのデータ処理  
アプリケーションに、オフラインとされた加速器 44 のディスエーブルされた区分の動作  
の「引き継ぎ」を行わせ得る。処理ユニット 62 (図 3) がこの動作を、加速器 44 より  
もより緩慢に且つより非効率的に実行し得るが、これはその動作を全く実行しないことよ  
りもより好ましい可能性がある。加速器 44 から処理ユニット 62 に動作の実行をシフト  
するこの能力は、ピアーベクトル・マシン 40 (図 3) の柔軟性、信頼性、保守性、並び  
に、フォルトトレランスを増大する。

#### 【0090】

そして、コンフィギュレーション・マネージャ 84 に送信されると、例外取り扱い命令  
はそのコンフィギュレーション・マネージャに加速器 44 のハード・コンフィギュレー  
ションを変更させ得て、加速器がオフラインとされた誤動作している区分の動作の実行を続  
行できるように為す。例えば、もし加速器 44 が未使用区分を有せば、コンフィギュレー  
ション・マネージャ 84 は誤動作していた区分での動作を実行すべくその未使用区分を構  
成し得る。もし加速器 44 が未使用区分を有しなければ、コンフィギュレーション・マネ  
ージャ 84 は、誤動作している区分の第 2 動作を実行すべく、すなわち、誤動作している  
区分を引き継ぐべく、第 1 動作を現在実行している加速器の区分を再構成し得る。この技  
術は、第 1 動作が省略され得るが、第 2 動作が省略できない場合、或はデータ処理アプリ  
ケーション 80 が第 2 動作の場合よりも第 1 動作の実行により適合している場合に有用で  
あり得る。加速器 44 の一方の区分から加速器の他方の区分まで動作の実行をシフトする

40

50

そうした能力は、ビアーベクトル・マシン 40 (図 3) の柔軟性、信頼性、保守性、並びに、フォルトトレランスを増大する。

#### 【0091】

図 7 で参照されるようにコンフィギュレーション・マネージャ 84 は、ビアーベクトル・マシン 40 (図 3) の初期化中に加速器 44 のハード・コンフィギュレーションを規定するファームウェアをロードし、そして、図 6 と連携して先に議論されたように、本発明の実施例に従って例外に応じて加速器のハード・コンフィギュレーションを再規定するファームウェアをロードし得る。以下に議論されるように、コンフィギュレーション・マネージャ 84 は加速器 44 の設計及び変更の複雑性をしばしば低減し、ビアーベクトル・マシン 40 (図 3) のフォルトトレランス、信頼性、保守性、並びに、柔軟性を増大する。 10

#### 【0092】

ビアーベクトル・マシン 40 の初期化中、コンフィギュレーション・マネージャ 84 は加速器コンフィギュレーション・レジストリ 70 からコンフィギュレーション・データを受信し、そのコンフィギュレーション・データによって識別されたコンフィギュレーション・ファームウェアをロードする。コンフィギュレーション・データは、効果としては、ファームウェアをロードするためのコンフィギュレーション・マネージャ 84 への命令である。例えば、もし初期化された加速器 44 の区分が FFT を実行する場合には、コンフィギュレーション・データを設計して、マネージャ 84 によってロードされたファームウェアが加速器のその区分で FFT を具現化するように為す。結果として、ビアーベクトル・マシン 40 の初期化前にコンフィギュレーション・データを単に生成或は変更することによって、加速器 44 のハード・コンフィギュレーションを変更できる。コンフィギュレーション・データの生成及び変更がファームウェアを直に生成及び変更するよりもしばしば容易であるので (特にもしコンフィギュレーション・データがコンフィギュレーション・マネージャ 84 にライブラリから現行ファームウェアをロードさせれば)、コンフィギュレーション・マネージャ 84 は加速器 44 の設計及び変更の複雑性を典型的には低減する。 20

#### 【0093】

コンフィギュレーション・マネージャ 84 がコンフィギュレーション・データによって識別されたファームウェアをロードする前に、コンフィギュレーション・マネージャは加速器 44 がコンフィギュレーション・データによって規定されたコンフィギュレーションを支援し得るかを決定する。例えば、もしコンフィギュレーション・データがコンフィギュレーション・マネージャ 84 に加速器 44 の特定 PLIC (不図示) に対するファームウェアをロードさせるように命令すれば、コンフィギュレーション・マネージャ 84 はデータをロードする前にその PLIC が存在していることを確認する。もし PLIC が存在しなければ、コンフィギュレーション・マネージャ 84 は加速器 44 の初期化を停止して、加速器がそのコンフィギュレーションを支援しないことをオペレータに通知する。 30

#### 【0094】

コンフィギュレーション・マネージャ 84 が、加速器が規定されたコンフィギュレーションを支援することを確認した後、コンフィギュレーション・マネージャはファームウェアを加速器 44 にロードし、該加速器がそのファームウェアで、例えばそのファームウェアをファームウェア・メモリ 52 にロードすることによって、そのハード・コンフィギュレーションを設定する。典型的には、コンフィギュレーション・マネージャ 84 は、生成、構造、並びに、動作の点で図 5 のチャネル 104 と類似している 1 つ或はそれ以上のチャネル 104 を介して加速器 44 にファームウェアを送信する。またコンフィギュレーション・マネージャ 84 は 1 つ或はそれ以上のチャネル 104 を介して加速器 44 からデータをも受信し得る。例えば、加速器 44 はそのハード・コンフィギュレーションの成就した設定の確認をコンフィギュレーション・マネージャ 84 に送信し得る。 40

#### 【0095】

加速器 44 のハード・コンフィギュレーションが設定された後、コンフィギュレーション・マネージャ 84 は、図 6 と連携して先に議論されたように例外マネージャ 84 からの 50

例外取り扱い命令に応じて加速器のハード・コンフィギュレーションを設定し得る。例外取り扱い命令に応じて、コンフィギュレーション・マネージャ 84 はレジストリ 70 から適切なコンフィギュレーション・データをダウンロードし、そのコンフィギュレーション・データによって識別された再構成ファームウェアをロードして、そのファームウェアをチャンネル 104<sub>1</sub> を介して加速器 44 に送信する。コンフィギュレーション・マネージャ 84 はチャンネル 104<sub>1</sub> を介して加速器 44 から成就された再構成の確認を受信し得る。図 6 と連携して先に議論されたように、コンフィギュレーション・マネージャ 84 は、ライン 89 (図 4) を介して例外マネージャ 82 から直接的に、或は、チャンネル 104<sub>1</sub>, 104<sub>2</sub> を介して間接的に、例外取り扱い命令を受信し得る。

#### 【0096】

またコンフィギュレーション・マネージャ 84 は、図 6 と連携して先に議論されたように、例外マネージャ 84 からの例外取り扱い命令に応じてデータ処理アプリケーション 80 を再構成し得る。例外取り扱い命令に応じて、コンフィギュレーション・マネージャ 84 はデータ処理アプリケーション 80 に対して、誤動作或は他の理由のため、加速器 44 が実行することができない動作を実行させるべく、それ自体を再構成するように命令する。コンフィギュレーション・マネージャ 84 は、ライン 87 (図 4) を介して直接的に、或はチャンネル 104<sub>1</sub>, 104<sub>2</sub> を介して間接的にデータ処理アプリケーション 80 にそのように命令し得て、直接的に或は別のチャンネル 104 (不図示) を介して、成就した再構成の確認等の情報をデータ処理アプリケーションから受信し得る。代替的には、例外マネージャ 82 は例外取り扱い命令をデータ処理 80 に送信し得て、該データ処理 80 はそれ自体を再構成し、よってコンフィギュレーション・マネージャ 82 を迂回する。

#### 【0097】

更に図 7 で参照されるように、コンフィギュレーション・マネージャ 82 の代替実施例が想定されている。例えば、コンフィギュレーション・マネージャ 82 は、加速器誤動作の発生以外の理由で、加速器 44 或はデータ処理アプリケーション 80 を再構成し得る。

#### 【0098】

先行する議論は当業者が本発明を作製し使用することを可能とすべく提示されている。種々実施例への様々な変更は当業者には容易に明かであろうし、ここでの包括的な原則は本発明の精神及び範囲から逸脱することなしに他の実施例及び適用例に適用され得る。よって、本発明は図示された実施例に限定されることが意図されておらず、ここに開示された原理及び特徴と一貫した最も広い範囲と一致されるべきものである。

#### 【図面の簡単な説明】

#### 【0099】

【図 1】図 1 は、従来の多数プロセッサ・アーキテクチャを有する計算マシンのブロック線図である。

【図 2】図 2 は、従来のハードウェアに組み込まれたパイプラインのブロック線図である。

【図 3】図 3 は、本発明の実施例に従ったビアーベクトル・アーキテクチャを有する計算マシンの概略ブロック線図である。

【図 4】図 4 は、本発明の実施例に従った図 3 のホストプロセッサの機能的ブロック線図である。

【図 5】図 5 は、本発明の実施例に従った図 4 のデータ処理アプリケーションとパイプライン・バスの間におけるデータ転送経路の機能的ブロック線図である。

【図 6】図 6 は、本発明の実施例に従った図 4 の加速器例外マネージャとパイプライン・バスの間におけるデータ転送経路の機能的ブロック線図である。

【図 7】図 7 は、本発明の実施例に従った図 4 の加速器コンフィギュレーション・マネージャとパイプライン・バスの間におけるデータ転送経路の機能的ブロック線図である。

#### 【符号の説明】

#### 【0100】

10 計算マシン

10

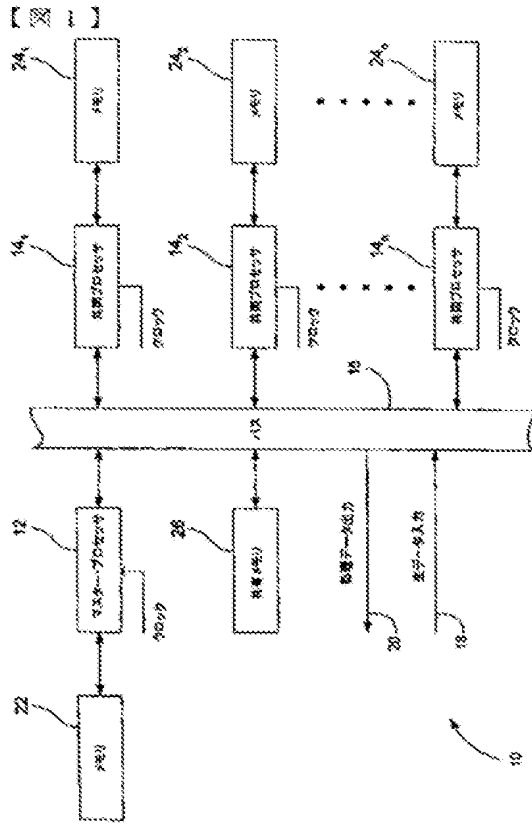
20

30

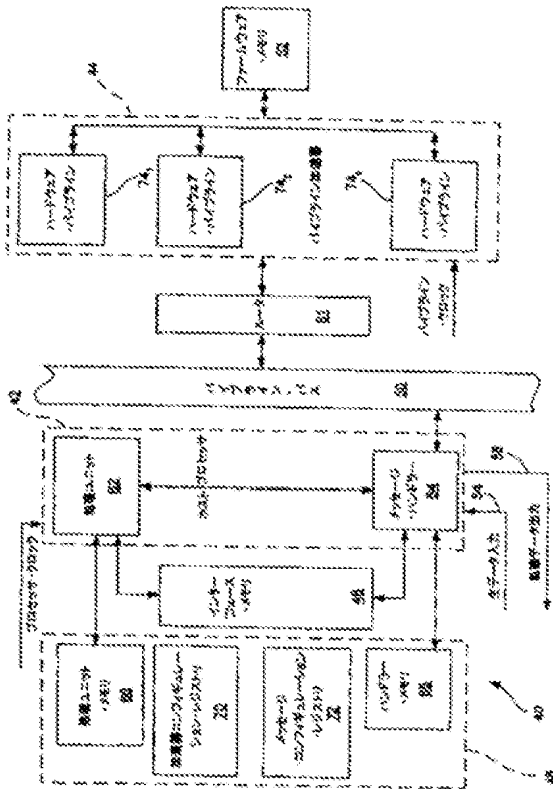
40

50

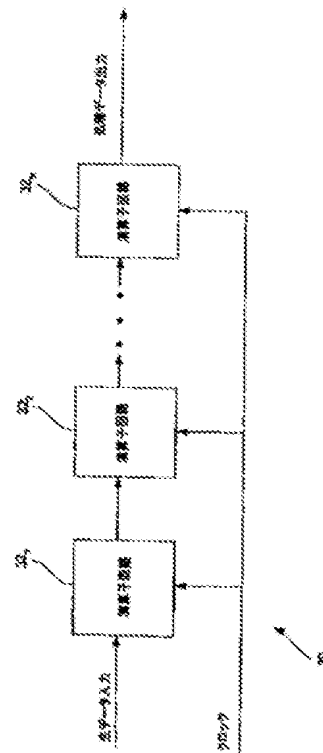
1 4	共同プロセッサ	
4 0	ビアーベクトル・マシン	
4 2	ホストプロセッサ	
4 4	パイプライン加速器	
4 6	プロセッサ・メモリ	
4 8	インターフェース・メモリ	
5 0	パイプライン・バス	
5 2	ファームウェア・メモリ	
5 4	生データ入力ポート	
5 8	処理済みデータ出力ポート	10
6 1	ルータ	
6 2	処理ユニット	
6 4	メッセージ・ハンドラー	
6 6	処理ユニット・メモリ	
6 8	ハンドラー・メモリ	
7 0	加速器コンフィギュレーション・レジストリ	
7 2	メッセージ・コンフィギュレーション・レジストリ	
7 4	ハードウェアに組み込まれたパイプライン	
7 8	パイプライン・ユニット	
8 0	データ処理アプリケーション	20
8 2	加速器例外マネージャ	
8 4	加速器コンフィギュレーション・マネージャ	
8 6	データ転送オブジェクト	
8 8	通信オブジェクト	
9 0	入力リーダー	
9 2	出力リーダー	
9 4	入力キュー	
9 6	出力キュー	
9 8	オブジェクト・ファクトリー	
1 0 0, - 1 0 0。	スレッド	30
1 0 6, - 1 0 6。	パッファ	



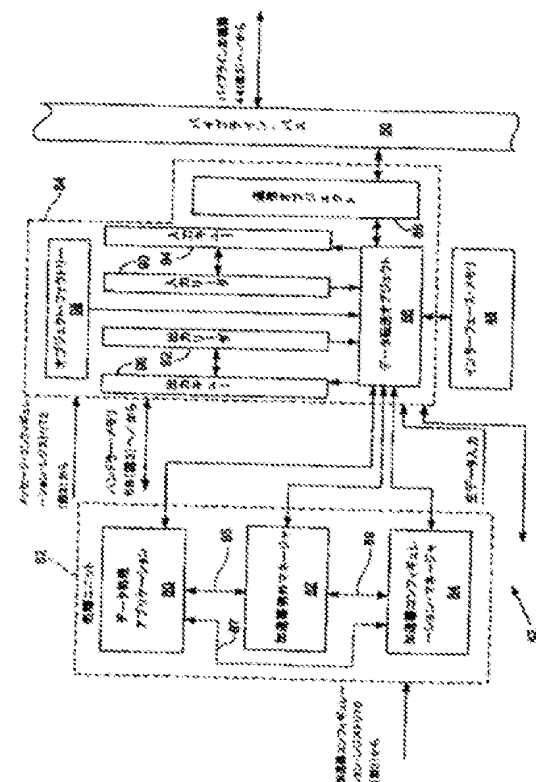
【図 3】



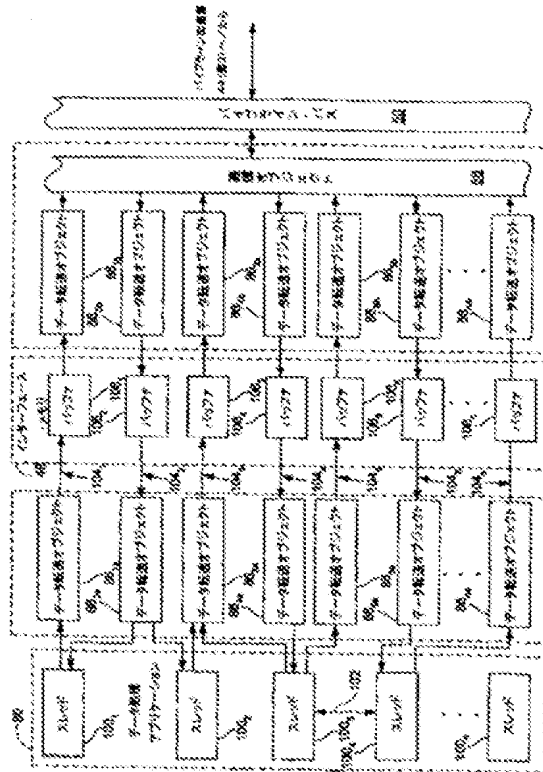
【図 2】



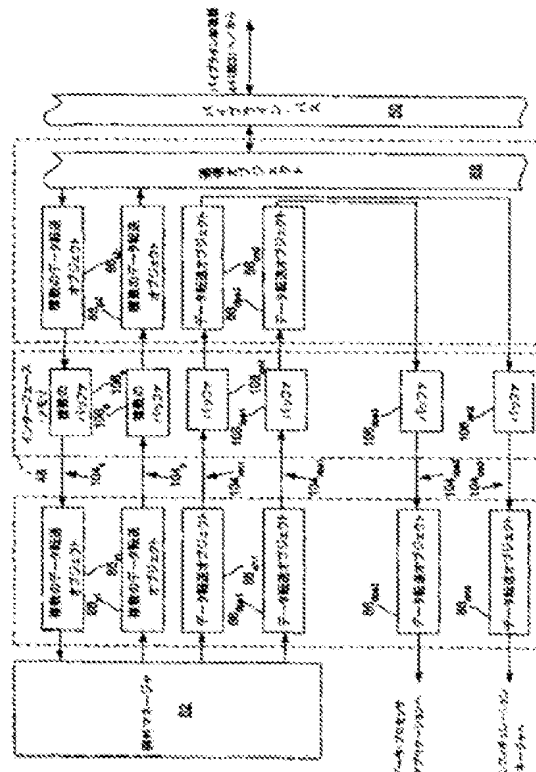
【図 4】



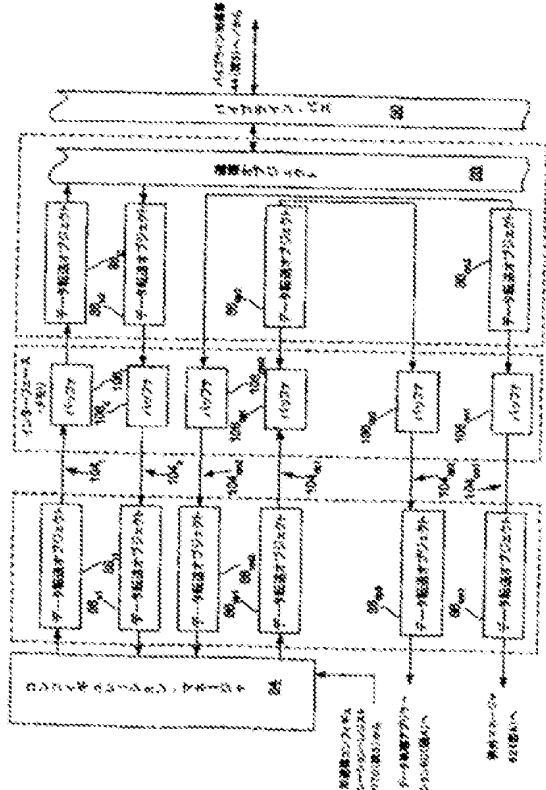
【図 5】



【図 6】



【図 7】



【国際調査報告】

INTERNATIONAL SEARCH REPORT		Inventor's Application No. PCT/US 03/34559
<b>A. CLASSIFICATION OF SUBJECT MATTER</b> IPC 7 0606F/46		
According to International Patent Classification (IPC) or to both national classification and IPC		
<b>B. FIELDS SEARCHED</b> Minimum documentation searched (classification system followed by classification symbols) IPC 7 0606F		
Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched		
Electronic data base consulted during the international search (name of data base and, where practicable, search terms used) EPO-Internal, INSPEC		
<b>C. DOCUMENTS CONSIDERED TO BE RELEVANT</b>		
Category *	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to state of the art
A	EP 0 694 847 A (IBM) 31 January 1996 (1996-01-31) column 1, line 1 - column 5, line 50	1-61
A	US 4 956 771 A (NEUSTAEDTER TARL) 11 September 1990 (1990-09-11) abstract column 1, line 12 - column 3, line 14	1-61
<input type="checkbox"/> Further documents are listed in the configuration of box C. <input checked="" type="checkbox"/> Patent family members are listed to ensure.		
* Special categories of cited documents : "A" document defining the general state of the art which is not considered to be of particular relevance "E" earlier document but published on or after the international filing date "L" document which may show double on priority claims or which is cited to establish the publication date of another claim or other special reason (as specified) "O" document referring to an oral disclosure, use, exhibition or other means "P" document published prior to the international filing date but later than the priority date claimed "T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention "X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone "Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art "Z" document member of the same patent family		
Date of the actual completion of the international search		Date of mailing of the international search report
23 December 2004		07/01/2005
Name and mailing address of the ISA European Patent Office, P.O. Box 1600, 7000 Lausanne, Switzerland Tel. (+41-22) 365 3140-2140, Tx. ST 851 epo nl Fax: (+41-22) 365 3015		Authorized officer Brandt, J

Form PCT/ISA/210 (second sheet) (January 2004)



## INTERNATIONAL SEARCH REPORT

Information on patent family members

In International Application No.

PCT/US 03/34559

Patent document cited in search report	Publication class	Publication date	Patent family member(s)	Publication date
EP 0694847	A	31-01-1996	US 5568614 A	22-10-1996
			CA 2152984 A1	30-01-1996
			EP 0694847 A2	31-01-1996
			JP 3251815 B2	28-01-2002
			JP 8055075 A	27-02-1996
			KR 163234 B1	15-01-1999
US 4956771	A	11-09-1990	NONE	

Form PCT/IS602.10 (patent family sheet) January 2004

## フロントページの続き

(31)優先権主張番号 10/683,932

(32)優先日 平成15年10月9日(2003.10.9)

(33)優先権主張国 米国(US)

(31)優先権主張番号 10/684,053

(32)優先日 平成15年10月9日(2003.10.9)

(33)優先権主張国 米国(US)

(31)優先権主張番号 10/684,057

(32)優先日 平成15年10月9日(2003.10.9)

(33)優先権主張国 米国(US)

(81)指定国 AP(BW, GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), EA(AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), EP(AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OA(BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG), AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BW, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, ES, FI, GB, GD, GE, GH, GM, GR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW

(74)代理人 100135585

弁理士 西尾 務

(72)発明者 マートウル, チャンダン

アメリカ合衆国 バージニア州 20109 マナサッス, プライベータス コート 1116  
2

(72)発明者 ヘレンバツハ, スコット

アメリカ合衆国 バージニア州 20106 アメッスビル, クアイル リッジ ドライブ 15  
381

(72)発明者 ラーブ, ジョン, ダブリュ.

アメリカ合衆国 バージニア州 20110 マナサッス, リバー クレスト ロード 9350